Best Practices for Storing the Esri® Production Mapping Workspace in an Enterprise Geodatabase for SQL Server

An Esri® Technical Paper
October 2012

esri

380 New York Street
Redlands, California 92373-8100 USA

909 793 2853
info@esri.com

esri.com

Best Practices for Storing the Esri Production Mapping Workspace in an Enterprise Geodatabase for SQL Server

An Esri Technical Paper

Contents **Page**

Best Practices for Storing the Esri Production Mapping Workspace in an Enterprise Geodatabase for SQL Server

Introduction

Esri® Production Mapping extends ArcGIS® to streamline your geographic information system (GIS) data and map production by providing tools that facilitate data creation, maintenance, and validation, as well as tools for producing high-quality cartographic products. Each organization has workflows that are unique to the type of data being collected and the type of product being delivered. These workflows can be generalized into a basic production workflow that consists of steps to create your geodatabase and capture or load an initial set of data, perform edits to the data, ensure the data is valid and accurate, and produce digital or hard-copy output. Production Mapping is designed to streamline each of these steps while remaining flexible to adapt to your business rules and workflows. This technical paper is intended to help database administrators establish the Production Mapping workspace in an enterprise geodatabase for SQL Server®. The enterprise geodatabase uses ArcSDE® technology as the gateway between GIS clients and SQL Server.

The Geodatabase

Central to any production workflow is the geodatabase. How you choose to design your geodatabase determines what data needs to be captured and edited, what logical rules exist for validating the data, and the types of output that can be created.

When designing a geodatabase, certain types of validation are built in, such as the ability to limit the valid attribute values for a field by using a domain. You can also model the geometric relationships of features through topologies or networks. Production Mapping supports these forms of validation and allows you to define additional business rules using ArcGIS Data Reviewer checks stored in a batch job. The batch jobs can be run when you update the attributes of a feature or template, like domains, or as a postprocess such as validating the topology.

The product library in Production Mapping provides a framework for managing business rules, data, and map documents in a secure, centralized location. By leveraging the rules stored in the product library, data editing tools are tailored to ensure that features conform to your data collection standards. During cartographic production, the product library can act as a management system for your map documents and data, allowing you to check in and check out files and restore historical versions. The cartographic tools in Production Mapping can also be used to ensure consistent and repeatable symbology as well as provide a number of dynamic surround elements.

See the white paper *Best Practices for Storing the Product Library Workspace in an Enterprise Geodatabase for SQL Server*.

Production Database

A production database contains the data used for production tasks such as creating and updating features. Depending on the data model you are using, data in a production database can be used to create a digital or hard-copy map/chart or a specific type of data. The data in a production database usually corresponds with a data model and product class in the product library. This technical paper is intended to help you establish the production database in an enterprise geodatabase for SQL Server.

Capture

The purpose of data capture is to consume existing data or create new data in your geodatabase. Data can be captured in many different ways, such as extracting new data from imagery, gathering information in the field with a mobile device, or converting existing data.

Production Mapping provides data loading automation tools that allow you to define the mapping ahead of time between source data, such as shapefiles or coverages, and your geodatabase to ensure consistency when converting large amounts of data. Feature Manager allows you to quickly and easily create new features by using feature templates and construction tools within a centralized editing environment that leverages your enhanced validation rules.

Editing

The editing stage of the production workflow involves adjusting existing features and adding new features to ensure that your data is up-to-date and accurate. This can involve updating data to match a new source or modifying attributes of data that was collected for a different purpose.

When editing data within Production Mapping, feature attribution is managed through Feature Manager, which allows you to update fields while ensuring the attributes are valid in accordance with your validation rules as well as update attributes in batches and create new features. Production Mapping also provides a number of editing tools for batch geometry updates, as well as tools for specific types of data or industries, including tools for linear referencing, utilities, contour lines, and z-enabled data.

Review

Data review is important to ensure that the data being created is accurate and appropriate for its intended purpose before it is delivered or used for making a map product. The data review or quality control stage of the production workflow often involves three phases: finding issues with the data, whether through manual or automated methods; fixing issues or marking them as exceptions; and verifying that the issues are resolved.

The Data Reviewer component of Production Mapping provides the ability to track records of all issues found in the Reviewer table, which maintains the status of the record as well as a link between the record and the feature with the issue. Data Reviewer offers the ability to automate data validation through configurable checks that can be run when you choose, scheduled through a Microsoft® Windows® service to run at specified times, or run through Python scripts. Automated checks may not be able to find all problems with the data, such as missing features. Data Reviewer also has a number of tools to help with manual or visual quality control, including tools to flag missing features, and the ability to create a sample set of data for in-depth validation.

Note: Data Reviewer is available with Production Mapping or as a stand-alone extension.

See the white paper *Best Practices for Storing the ArcGIS Data Reviewer Workspace in an Enterprise Geodatabase for SQL Server*.

Create Output

Typically, the final stage of a production workflow is to create the output that will be delivered; however, the workflow may be repeated if you're making a delivery contingent on approval or creating a product that requires regular updates. There are many types of output that may be produced: data exported to a certain format, hard-copy maps, or data that will be served over the web.

When producing hard-copy products, there is a wide range of types of maps or charts that you may need to produce, from one-off maps to map books to highly detailed charts that require version tracking. Production Mapping provides support for hard-copy map production through the product library by providing check-in and checkout capabilities for map documents as well as the ability to track history and roll back to previous versions.

Data visualization and symbology are important when producing a hard-copy product or serving data over the web. Production Mapping provides tools for consistent, repeatable, rule-based symbology, where you define what symbol or representation should be applied to features based on their attribute combination. Production Mapping also provides a number of custom surround elements, such as the graphic table element, which allows you to create a table or legend that automatically updates based on the data being displayed.

Workflow Management

When managing production, it is important to be able to allocate resources and track the status of the project. Being able to ensure that work is being done consistently and steps are not being skipped is also essential. Production Mapping allows you to tie all the components of data capture, editing, validation, and cartography together in high-level workflows with ArcGIS Workflow Manager and in detailed workflows with Task Assistant Manager.

Note: ArcGIS Workflow Manager is available with Production Mapping and as a stand-alone extension.

Workflow Manager allows you to create a job (unit of work), assign or reassign the job to a user, and track the overall status of the project. Each job includes a predefined workflow that you build to represent your processes. When assigned a job, you execute the steps in the workflow to launch the appropriate tools, send e-mail notifications, or ask questions to determine the path the job should take.

Task Assistant Manager allows you to define workflows in ArcMap™ that guide you through various tasks. Task Assistant workflows can be used to provide step-by-step instructions for complex tasks, to minimize confusion for new users, or simply as a reference. Clicking a task in a workflow can execute either a tool in ArcMap or geoprocessing tools, set up your environment by specifying layer or snapping properties, or provide a description of what needs to be done.

See the white paper *Best Practices for Storing the ArcGIS Workflow Manager Workspace in an Enterprise Geodatabase for SQL Server.*

Mapping and Charting Solutions

Production Mapping can be used to build your own solutions by creating a product library to meet the standards of an industry or business. However, Esri has built three commercial off-the-shelf solutions for the defense mapping, nautical, and aeronautical industries that utilize and expand on Production Mapping functionality.

ArcSDE DBTUNE

DBTUNE storage parameters let you control how ArcSDE technology creates objects within a SQL Server database. You can determine things such as how to allocate space to a table or index, the FileGroup that a table or index is created in, and other SQL Server-specific storage attributes. They also allow you to specify one of the available storage formats for the geometry of a spatial column.

The DBTUNE storage parameters are stored in the DBTUNE table, which—along with all other metadata tables—is created during the setup phase that follows the installation of ArcSDE. ArcSDE installation creates a dbtune file under the etc directory from which the DBTUNE table is populated. If no dbtune file is present during setup, the DBTUNE table will be populated with default values.

When a large number of database connections access the same files in the same location on disk, database performance is slower because the connections are competing with one another for the same resources. To reduce this competition, you can store database files in different locations on disk.

Thus, DBTUNE can be modified to store the Production feature dataset and tables in separate data files across different locations on disk. This will reduce disk contention and improve database input/output (I/O).

Standard GIS storage recommendations favor keeping index and log files separate from vector and tabular business tables. For performance reasons, it is better to position the business, feature, and spatial index tables separately and to position FileGroup data files based on their usage pattern. For a multiversioned, highly active editing geodatabase, database files of the VERSIONS FileGroup may be separated and dispersed across available disks to avoid I/O contention.

Disk Configuration

Large production enterprise geodatabase systems should employ a hardware striping solution. Your best disk and data organization strategies involve spreading your data across multiple disks.

When data is spread across multiple disks, more spindles can actively search for it. This can increase disk read time and decrease disk contention. However, too many disks can slow down a query. There are two main ways of achieving striping: FileGroups and redundant array of independent disks (RAID). You can also combine the two by creating FileGroups within disk arrays. You can employ data segregation strategies; keeping tables from indexes or certain types of tables from other tables will improve performance and alleviate administrative burdens.

The suggested SQL Server optimal configuration is as follows:

- Disk 0—SQL Server/Application software
- Disk 1—master, model, msdb
- Disk 2—tempdb
- Disk 3—Log files
- Disk 4—Feature data tables
- Disk 5—Spatial index data tables
- Disk 6—Attribute data/Business tables
- Disk 7—SQL Server indexes

Reducing Disk I/O Contention

As a rule, you should create database files that are as large as possible, based on the maximum amount of data you estimate the database will contain, to accommodate future growth. By creating large files, you can avoid file fragmentation and gain better database performance. In many cases, you can let data files grow automatically; just be sure to limit autogrowth by specifying a maximum growth size that leaves some hard disk space available. By putting different FileGroups on different disks, you can also minimize physical fragmentation of your files as they grow.

To configure data and log files for best performance, follow these best practices:

- To avoid disk contention, do not put data files on the drive that contains the operating system files.

- Put transaction log files and data files on separate drives. This will give you the best performance by reducing disk contention between data and transaction log files.

- Put the tempdb database on a separate drive if possible, preferably on a RAID 10 or RAID 5 system. In environments in which there is intensive use of tempdb databases, you can get better performance by putting tempdb on a separate drive, which would let SQL Server perform tempdb operations in parallel with database operations.

- Understand that the RAID configuration that is best for your database files depends on several factors, including performance and recoverability needs. RAID 10 is the recommended RAID system for transaction log, data, and index files. If you have budget restrictions, consider keeping transaction log files in a RAID 10 system and storing data and index files in a RAID 5 system.

  For more information about RAID, see RAID Levels and SQL Server at http://technet.microsoft.com/en-us/library/ms190764(SQL.105).aspx and chapter 7, "Planning Fault Tolerance and Avoidance," by Charlie Russel and Sharon Crawford, from *Microsoft Windows 2000 Server Administrator's Companion* (Microsoft Press) at http://technet.microsoft.com/en-us/library/bb742464.aspx.

- Use partitions on large tables. Partitioning lets you split a table across multiple FileGroups; you can place a subset of a table or index on a designated FileGroup. This capability lets you separate specific pieces of a table or index onto individual

FileGroups and effectively manage file I/O for volatile tables. Partitions let you easily manage archival routines and data loading operations.

Below is a suggested design to reduce disk I/O contention:

File Type	Database Activity	Move File to Disk With
Transaction log files	Frequent edits	Relatively low I/O
Transaction log files	Few or no edits	Moderate I/O
tempdb	Frequent edits	Low I/O but separate from transaction log files
master, model, msdb	Few edits	Moderate I/O
Data	Frequent edits	Relatively low I/O

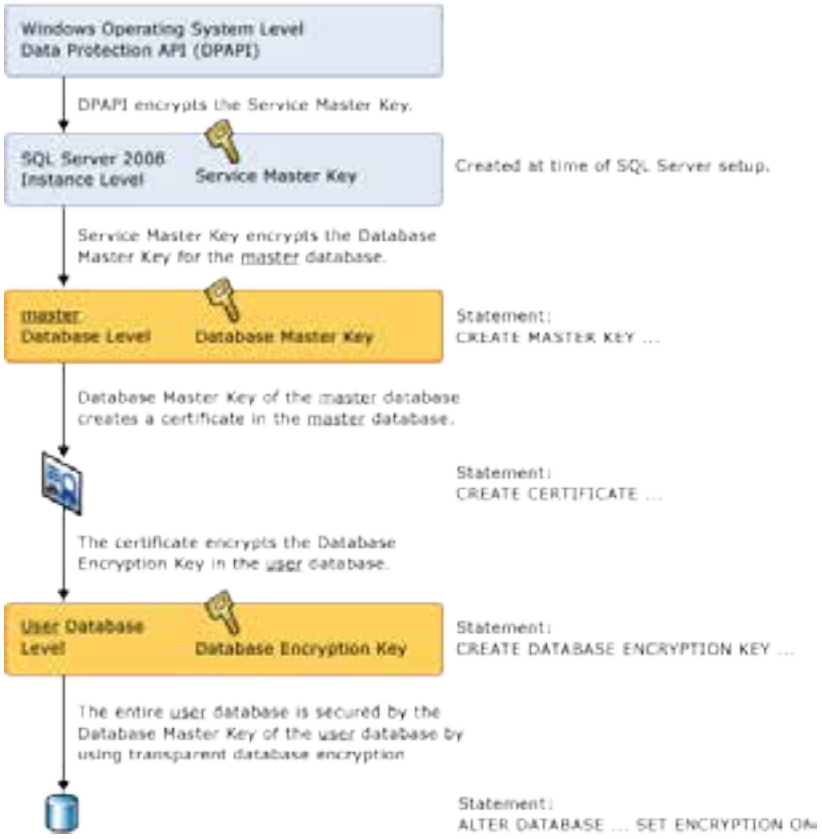
Transparent Data Encryption

The precautions you can take to help protect the database include designing a secure system, encrypting confidential assets, and building a firewall around the database servers. However, if the physical media (drives or backup tapes) are stolen, a malicious party could just restore or attach the database and browse the data. One solution is to encrypt sensitive data in the database and protect the keys that are used to encrypt the data with a certificate. This prevents anyone without the keys from using the data, but this kind of protection must be planned in advance.

Transparent data encryption (TDE) performs real-time I/O encryption and decryption of the data and log files. The encryption uses a database encryption key (DEK), which is stored in the database boot record for availability during recovery. The DEK is either a symmetric key secured by using a certificate stored in the master database of the server or an asymmetric key protected by an extensible key management (EKM) module. TDE protects data "at rest," meaning the data and log files. It provides the ability to comply with many laws, regulations, and guidelines established in various industries. This enables software developers to encrypt data by using Advanced Encryption Standard (AES) and Triple Data Encryption Standard (3DES) encryption algorithms without changing existing applications.

Database files are encrypted at the page level. The pages are encrypted before they are written to disk and decrypted when read into memory. TDE does not increase the size of the encrypted database.

The following illustration shows the architecture of TDE encryption:



TDE Encryption Architecture

Learn more about TDE at
http://msdn.microsoft.com/en-us/library/bb934049.aspx.

To use TDE, follow these steps:

- Create a master key.
- Create or obtain a certificate protected by the master key.
- Create a database encryption key and protect it with the certificate.
- Set the database to use encryption.

```
USE master
GO
/* Verify master key */
SELECT * FROM sys.symmetric_keys WHERE name LIKE '%MS_DatabaseMasterKey%'
GO

/* if there are no records found, then it means there was no predefined Master Key.
 To create a Master Key, you can execute the below mentioned TSQL code. */

/* Create master key */
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'pmadmin';
GO
/* Backup master key */
OPEN MASTER KEY DECRYPTION BY PASSWORD = 'pmadmin';
GO
BACKUP MASTER KEY TO FILE = 'D:\mssqlbackup\master\masterkey.mk'
    ENCRYPTION BY PASSWORD = 'pmadmin';
GO

/* Create Certificate */
CREATE CERTIFICATE pm_cert WITH SUBJECT = 'PM Server Certificate';
GO

/* Verify Certificate */
SELECT * FROM sys.certificates where [name] = 'pm_cert'
GO

/* Backup certificate */
BACKUP CERTIFICATE pm_cert TO FILE = 'D:\mssqlbackup\master\pm.cer'
   WITH PRIVATE KEY (
         FILE = 'D:\mssqlbackup\master\pm.pvk',
         ENCRYPTION BY PASSWORD = 'pmadmin');
GO

USE pmdb
GO
/* Create Encryption key */
CREATE DATABASE ENCRYPTION KEY
   WITH ALGORITHM = AES_256
   ENCRYPTION BY SERVER CERTIFICATE pm_cert;
GO

/* Encrypt database */
ALTER DATABASE pmdb SET ENCRYPTION ON;
GO

/* Verify Encryption */
SELECT
DB_NAME(database_id) AS DatabaseName
,Encryption_State AS EncryptionState
,key_algorithm AS Algorithm
,key_length AS KeyLength
FROM sys.dm_database_encryption_keys
GO
SELECT
NAME AS DatabaseName
,IS_ENCRYPTED AS IsEncrypted
FROM sys.databases where name ='pmdb'
GO
```

Step 1: Create Data Files

Create new file groups to store the production data.

```
FILEGROUP         ArcSDE_PARAMETER
PM_BDATA       PM_Bdata       Business table
PM_BINDEX      PM_Bindex      Business table index
PM_BDATA_TOPO  PM_Bdata_Topo  Topology Business table
PM_BINDEX_TOPO PM_Bindex_Topo Topology Business table index
PM_FDATA       PM_FDATA       ST_Geomery Lob storage
PM_FINDEX      PM_FINDEX      ST_Geomery Lob index
PM_FDATA_TOPO  PM_FDATA_Topo  Topology ST_Geomery Lob storage
PM_FINDEX_TOPO PM_FINDEX_Topo Topology ST_Geomery Lob index
PM_SDATA       PM_Sdata       Spatial Index Tables
PM_SINDEX      PM_Sindex      Spatial Index Features
PM_SDATA_TOPO  PM_Sdata_Topo  Topology Spatial Index Tables
PM_SINDEX_TOPO PM_Sindex_Topo Topology Spatial Index Features
PM_ADATA       PM_Adata       Adds table (versioned)
PM_AINDEX      PM_Aindex      Adds table index
PM_DDATA       PM_Ddata       Delete table (versioned)
PM_DINDEX      PM_Dindex      Delete table index
PM_RASTER      PM_raster      Raster Datasets
PM_RINDEX      PM_rindex      Raster Indexes
PM_RBLK        PM_rblk        Raster BLK table
PM_RBLKIDX     PM_rblkidx     Raster BLK table Indexes
PM_XMLDOC      PM_xmldoc      XML Documents
```

J10067

```
PM_XMLIDX      PM_xmlidx      XML Documents Indexes

USE MASTER
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_BDATA]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Bdata01', FILENAME =
N'C:\mssql\data\pmdb\pm_Bdata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_BDATA]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_BINDEX]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Bindex01', FILENAME =
N'C:\mssql\data\pmdb\pm_Bindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_BINDEX]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_BDATA_TOPO]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Bdata_topo01', FILENAME =
N'C:\mssql\data\pmdb\pm_Bdata_topo01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_BDATA_TOPO]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_BINDEX_TOPO]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Bindex_topo01', FILENAME =
N'C:\mssql\data\pmdb\pm_Bindex_topo01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_BINDEX_TOPO]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_FDATA]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Fdata01', FILENAME =
N'C:\mssql\data\pmdb\pm_Fdata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_FDATA]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_FINDEX]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Findex01', FILENAME =
N'C:\mssql\data\pmdb\pm_Findex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_FINDEX]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_FDATA_TOPO]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Fdata_topo01', FILENAME =
N'C:\mssql\data\pmdb\pm_Fdata_topo01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_FDATA_TOPO]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_FINDEX_TOPO]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Findex_topo01', FILENAME =
N'C:\mssql\data\pmdb\pm_Findex_topo01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_FINDEX_TOPO]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_SDATA]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Sdata01', FILENAME =
N'C:\mssql\data\pmdb\pm_Sdata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_SDATA]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_SINDEX]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Sindex01', FILENAME =
N'C:\mssql\data\pmdb\pm_Sindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_SINDEX]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_SDATA_TOPO]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Sdata_topo01', FILENAME =
N'C:\mssql\data\pmdb\pm_Sdata_topo01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_SDATA_TOPO]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_SINDEX_TOPO]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Sindex_topo01', FILENAME =
N'C:\mssql\data\pmdb\pm_Sindex_topo01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_SINDEX_TOPO]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_ADATA]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Adata01', FILENAME =
N'C:\mssql\data\pmdb\pm_Adata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_ADATA]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_AINDEX]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Aindex01', FILENAME =
N'C:\mssql\data\pmdb\pm_Aindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_AINDEX]
GO
```

```
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_DDATA]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Ddata01', FILENAME =
N'C:\mssql\data\pmdb\pm_Ddata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_DDATA]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_DINDEX]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Dindex01', FILENAME =
N'C:\mssql\data\pmdb\pm_Dindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_DINDEX]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_RASTER]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Raster01', FILENAME =
N'C:\mssql\data\pmdb\pm_Raster01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_RASTER]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_RINDEX]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Rindex01', FILENAME =
N'C:\mssql\data\pmdb\pm_Rindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_RINDEX]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_RBLK]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Rblk01', FILENAME = N'C:\mssql\data\pmdb\pm_Rblk01.NDF'
, SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP [PM_RBLK]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_RBLKIDX]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_Rblkidx01', FILENAME =
N'C:\mssql\data\pmdb\pm_Rblkidx01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_RBLKIDX]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_XMLDOC]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_xmldoc01', FILENAME =
N'C:\mssql\data\pmdb\pm_xmldoc01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_XMLDOC]
GO
ALTER DATABASE [PMDB] ADD FILEGROUP [PM_XMLIDX]
GO
ALTER DATABASE [PMDB] ADD FILE(NAME = N'pm_xmlidx01', FILENAME =
N'C:\mssql\data\pmdb\pm_xmlidx01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[PM_XMLIDX]
GO
```

By setting the data files' initial size to 1 MB, there is no delay in the creation of the FileGroups; to avoid fragmentation, you can resize the data files.

```
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Bdata01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Bindex01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Bdata_topo01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Bindex_topo01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Fdata01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Findex01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Fdata_topo01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Findex_topo01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Sdata01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Sindex01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Sdata_topo01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Sindex_topo01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Adata01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Aindex01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Ddata01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Dindex01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Raster01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Rindex01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Rblk01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_Rblkidx01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_xmldoc01’, SIZE = 400MB )
ALTER DATABASE [PMDB] MODIFY FILE ( NAME = N’pm_xmlidx01’, SIZE = 400MB )
```

Verify FileGroups and data files:

```
EXEC sp_helpdb pmdb
GO
```

Step 2: Create PM User

Create a new database user to store the production data, then grant the appropriate permissions.

Create user and schema:

```
USE master
GO
CREATE LOGIN pm WITH PASSWORD = 'pmadmin', DEFAULT_DATABASE=[pmdb],DEFAULT_LANGUAGE=[us_english],
CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
GO
USE PMDB
GO
CREATE USER [pm] FOR LOGIN [pm]
GO
CREATE SCHEMA [pm] AUTHORIZATION [pm]
GO
ALTER USER [pm] WITH DEFAULT_SCHEMA=[pm]
GO
```

Grant privileges:

```
USE PMDB
GO
EXEC sp_droprolemember 'gis_data_creator', 'pm'
GO
EXEC sp_droprole 'gis_data_creator'
GO
CREATE ROLE gis_data_creator AUTHORIZATION dbo
GO
GRANT CREATE TABLE TO gis_data_creator
GO
GRANT CREATE PROCEDURE TO gis_data_creator
GO
GRANT CREATE VIEW TO gis_data_creator
GO
EXEC sp_addrolemember 'gis_data_creator', 'pm'
GO
```

Verify role:

```
EXEC sp_helprolemember 'gis_data_creator'
GO
```

Verify role permissions:

```
select dp.NAME AS principal_name,
 dp.type_desc AS principal_type_desc,
 o.NAME AS object_name,
 p.permission_name,
 p.state_desc AS permission_state_desc
 from sys.database_permissions p
 left OUTER JOIN sys.all_objects o
 on p.major_id = o.OBJECT_ID
 inner JOIN sys.database_principals dp
 on p.grantee_principal_id = dp.principal_id
 where dp.NAME = 'gis_data_creator'
GO
```

Verify user permissions:

```
select USER_NAME(p.grantee_principal_id) AS principal_name,
 dp.type_desc AS principal_type_desc,
 p.class_desc,
 OBJECT_NAME(p.major_id) AS object_name,
 p.permission_name,
 p.state_desc AS permission_state_desc
from sys.database_permissions p
inner JOIN sys.database_principals dp
on p.grantee_principal_id = dp.principal_id
where USER_NAME(p.grantee_principal_id) = 'pm'
```

Associate login PM with user PM:

```
USE PMDB
GO
EXEC sp_change_users_login 'update_one','pm','pm'
GO
EXEC sp_helpuser 'pm'
GO
```

Step 3: Modify DBTUNE

Export the dbtune file before making any modification.

```
sdedbtune -o export -f dbtune_exp.sde -u sde -p sde -i sde:sqlserver:mcsdbsrv -s mcsdbsrv –D pmdb
```

Copy ***dbtune_exp.sde*** to ***dbtune_pm.sde***.

Fill the new dbtune file with the proper FileGroups:

```
dbtune_pm.sde

##DATA_DICTIONARY
B_CLUSTER_ROWID           0
B_CLUSTER_USER            0
B_INDEX_ROWID             "WITH FILLFACTOR = 75 ON PM_SDE_DICT_INDEX"
B_INDEX_USER              "WITH FILLFACTOR = 75 ON PM_SDE_DICT_INDEX"
B_STORAGE                 "ON PM_SDE_DICT"
I_STORAGE                 "ON PM_SDE_DICT"
MVTABLES_MODIFIED_INDEX   "WITH FILLFACTOR = 75 ON PM_SDE_DICT_INDEX"
MVTABLES_MODIFIED_TABLE   "ON PM_SDE_DICT"
STATE_LINEAGES_INDEX      "WITH FILLFACTOR = 75 ON SDE_STATE_INDEX"
STATE_LINEAGES_TABLE      "ON PM_SDE_STATE"
STATES_INDEX              "WITH FILLFACTOR = 75 ON SDE_STATE_INDEX"
STATES_TABLE              "ON PM_SDE_STATE"
VERSIONS_INDEX            "WITH FILLFACTOR = 75 ON SDE_STATE_INDEX"
VERSIONS_TABLE            "ON PM_SDE_STATE"
XML_INDEX_TAGS_INDEX      "WITH FILLFACTOR = 75 ON SDE_DICT_INDEX"
XML_INDEX_TAGS_TABLE      "ON PM_SDE_DICT"
END

##DEFAULTS
A_CLUSTER_RASTER          0
A_CLUSTER_ROWID           0
A_CLUSTER_SHAPE           1
A_CLUSTER_STATEID         0
A_CLUSTER_USER            0
A_CLUSTER_XML             0
A_INDEX_RASTER            "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_ROWID             "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_SHAPE             "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_STATEID           "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_USER              "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_XML               "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_MS_SPINDEX              "GRIDS = (MEDIUM, MEDIUM, MEDIUM, MEDIUM), CELLS_PER_OBJECT = 16"
A_OUT_OF_ROW              0
A_STORAGE                 "ON PM_ADATA"
AUX_CLUSTER_COMPOSITE     1
AUX_INDEX_COMPOSITE       "WITH FILLFACTOR = 75 ON PM_RASTERIDX"
AUX_STORAGE               "ON PM_RASTER"
B_CLUSTER_RASTER          0
B_CLUSTER_ROWID           0
B_CLUSTER_SHAPE           1
B_CLUSTER_TO_DATE         0
B_CLUSTER_USER            0
B_CLUSTER_XML             0
B_INDEX_RASTER            "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_INDEX_ROWID             "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_INDEX_SHAPE             "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_INDEX_TO_DATE           "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_INDEX_USER              "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_INDEX_XML               "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_MS_SPINDEX              "GRIDS = (MEDIUM, MEDIUM, MEDIUM, MEDIUM), CELLS_PER_OBJECT = 16"
B_OUT_OF_ROW              0
B_STORAGE                 "ON PM_BDATA"
BLK_CLUSTER_COMPOSITE     1
BLK_INDEX_COMPOSITE       "WITH FILLFACTOR = 75 ON PM_RASTERIDX"
BLK_OUT_OF_ROW            0
BLK_STORAGE               "ON PM_RASTER"
BND_CLUSTER_COMPOSITE     0
BND_CLUSTER_ID            0
BND_INDEX_COMPOSITE       "WITH FILLFACTOR = 75 ON PM_RASTERIDX"
BND_INDEX_ID              "WITH FILLFACTOR = 75 ON PM_RASTERIDX"
BND_STORAGE               "ON PM_RASTER"
COLLATION_NAME            ""
CROSS_DB_QUERY_FILTER     0
D_CLUSTER_ALL             0
D_CLUSTER_DELETED_AT      1
D_INDEX_ALL               "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_INDEX_DELETED_AT        "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_STORAGE                 "ON PM_DDATA"
F_CLUSTER_FID             1
F_INDEX_AREA              "WITH FILLFACTOR = 75 ON PM_FINDEX"
```

J10067

##LOGFILE_DEFAULTS
SESSION_TEMP_TABLE parameter must be set to 1 (true) to allow the session-based log file table to be created in tempdb. in this case to not fill the DBTUNE filegroups.

Example:

```
##LOGFILE_DEFAULTS
LD_INDEX_ALL            "WITH FILLFACTOR = 75"
LD_STORAGE              ""
LF_CLUSTER_ID           0
LF_CLUSTER_NAME         0
LF_INDEX_ID             "WITH FILLFACTOR = 75 "
LF_INDEX_NAME           "WITH FILLFACTOR = 75 "
LF_STORAGE              "ON SDELOG"
SESSION_INDEX           "WITH FILLFACTOR = 75 "
SESSION_STORAGE         ""
SESSION_TEMP_TABLE      0
END
```

If you change the SESSION_TEMP_TABLE parameter to 0 (false), the SDE_LOGFILES, SDE_LOGFILE_DATA, and SDE_SESSION<SDE_ID> tables will be created in the connecting user's schema; hence, the user requires CREATE TABLE permission.

Example:

```
##LOGFILE_DEFAULTS
LD_INDEX_ALL            "WITH FILLFACTOR = 75
ON PM_SDELOGIDX"
LD_STORAGE              "ON PM_SDELOG"
LF_CLUSTER_ID           0
LF_CLUSTER_NAME         0
LF_INDEX_ID             "WITH FILLFACTOR = 75
ON PM_SDELOGIDX"
LF_INDEX_NAME           "WITH FILLFACTOR = 75
ON PM_SDELOGIDX"
LF_STORAGE              "ON SDELOG"
SESSION_INDEX           "WITH FILLFACTOR = 75
ON PM_SDELOGIDX"
SESSION_STORAGE         "ON PM_SDELOG"
SESSION_TEMP_TABLE      0
END
```

```
F_INDEX_FID                 "WITH FILLFACTOR = 75 ON PM_FINDEX"
F_INDEX_LEN                 "WITH FILLFACTOR = 75 ON PM_FINDEX"
F_OUT_OF_ROW                0
F_STORAGE                   "ON PM_FDATA"
GEOM_SRID_CHECK             1
GEOMETRY_STORAGE            "GEOMETRY"
GEOMTAB_OUT_OF_ROW          0
GEOMTAB_PK                  "WITH FILLFACTOR = 75 ON PM_FINDEX"
GEOMTAB_STORAGE             "ON PM_FDATA"
I_STORAGE                   "ON PM_FDATA"
NUM_DEFAULT_CURSORS         -1
PERMISSION_CACHE_THRESHOLD 250
RAS_CLUSTER_ID              1
RAS_INDEX_ID                "WITH FILLFACTOR = 75 ON PM_RASTERIDX"
RAS_STORAGE                 "ON PM_RASTER"
RASTER_STORAGE              "BINARY"
S_CLUSTER_ALL               1
S_CLUSTER_SP_FID            0
S_INDEX_ALL                 "WITH FILLFACTOR = 75 ON PM_SINDEX"
S_INDEX_SP_FID              "WITH FILLFACTOR = 75 ON PM_SINDEX"
S_STORAGE                   "ON PM_SDATA"
UI_TEXT                     "User Interface text for DEFAULTS"
UNICODE_STRING              "TRUE"
XML_COLUMN_STORAGE          "DB_XML"
XML_DOC_INDEX               "WITH FILLFACTOR = 75 ON PM_XMLIDX"
XML_DOC_OUT_OF_ROW          0
XML_DOC_STORAGE             "ON PM_XMLDOC"
XML_DOC_UNCOMPRESSED_TYPE   "BINARY"
XML_IDX_CLUSTER_DOUBLE      0
XML_IDX_CLUSTER_ID          0
XML_IDX_CLUSTER_PK          1
XML_IDX_CLUSTER_STRING      0
XML_IDX_CLUSTER_TAG         0
XML_IDX_FULLTEXT_CAT        "SDE_DEFAULT_CAT"
XML_IDX_FULLTEXT_LANGUAGE   ""
XML_IDX_FULLTEXT_TIMESTAMP  1
XML_IDX_FULLTEXT_UPDATE_METHOD "CHANGE_TRACKING BACKGROUND"
XML_IDX_INDEX_DOUBLE        "WITH FILLFACTOR = 75 ON PM_XMLIDX"
XML_IDX_INDEX_ID            "WITH FILLFACTOR = 75 ON PM_XMLIDX"
XML_IDX_INDEX_PK            "WITH FILLFACTOR = 75 ON PM_XMLIDX"
XML_IDX_INDEX_STRING        "WITH FILLFACTOR = 75 ON PM_XMLIDX"
XML_IDX_INDEX_TAG           "WITH FILLFACTOR = 75 ON PM_XMLIDX"
XML_IDX_OUT_OF_ROW          0
XML_IDX_STORAGE             "ON PM_XMLIDX"
END

##LOGFILE_DEFAULTS
LD_INDEX_ALL                "WITH FILLFACTOR = 75 ON PM_SDELOGIDX"
LD_STORAGE                  "ON PM_SDELOG"
LF_CLUSTER_ID               0
LF_CLUSTER_NAME             0
LF_INDEX_ID                 "WITH FILLFACTOR = 75 ON PM_SDELOGIDX"
LF_INDEX_NAME               "WITH FILLFACTOR = 75 ON PM_SDELOGIDX"
LF_STORAGE                  "ON SDELOG"
SESSION_INDEX               "WITH FILLFACTOR = 75 ON PM_SDELOGIDX"
SESSION_STORAGE             "ON PM_SDELOG"
SESSION_TEMP_TABLE          0
END

##GEOGRAPHY
GEOMETRY_STORAGE            "GEOGRAPHY"
UI_TEXT                     "User Interface text description for GEOGRAPHY"
END

##GEOMETRY
GEOMETRY_STORAGE            "GEOMETRY"
UI_TEXT                     "User Interface text description for GEOMETRY"
END

##IMS_GAZETTEER
XML_COLUMN_STORAGE          "SDE_XML"
END

##NETWORK_DEFAULTS
A_CLUSTER_ROWID             0
A_CLUSTER_SHAPE             1
A_CLUSTER_STATEID           0
A_CLUSTER_USER              0
A_INDEX_ROWID               "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_SHAPE               "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_STATEID             "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_USER                "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_OUT_OF_ROW                0
A_STORAGE                   "ON PM_ADATA"
B_CLUSTER_ROWID             0
B_CLUSTER_SHAPE             1
B_CLUSTER_USER              0
```

```
B_INDEX_ROWID              "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_INDEX_SHAPE              "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_INDEX_USER               "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_OUT_OF_ROW               0
B_STORAGE                  "ON PM_BDATA"
COMMENT                    "The base system initialization parameters for NETWORK_DEFAULTS"
D_CLUSTER_ALL              0
D_CLUSTER_DELETED_AT       1
D_INDEX_ALL                "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_INDEX_DELETED_AT         "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_STORAGE                  "ON PM_DDATA"
F_CLUSTER_FID              1
F_INDEX_AREA               "WITH FILLFACTOR = 75 ON PM_FINDEX"
F_INDEX_FID                "WITH FILLFACTOR = 75 ON PM_FINDEX"
F_INDEX_LEN                "WITH FILLFACTOR = 75 ON PM_FINDEX"
F_OUT_OF_ROW               0
F_STORAGE                  "ON PM_FDATA"
S_CLUSTER_ALL              1
S_CLUSTER_SP_FID           0
S_INDEX_ALL                "WITH FILLFACTOR = 75 ON PM_SINDEX"
S_INDEX_SP_FID             "WITH FILLFACTOR = 75 ON PM_SINDEX"
S_STORAGE                  "ON PM_SDATA"
UI_NETWORK_TEXT            "The network default configuration"
END

##NETWORK_DEFAULTS::DESC
A_CLUSTER_ROWID            1
A_CLUSTER_STATEID          0
A_CLUSTER_USER             0
A_INDEX_ROWID              "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_STATEID            "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_USER               "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_OUT_OF_ROW               0
A_STORAGE                  "ON PM_ADATA"
B_CLUSTER_ROWID            1
B_CLUSTER_USER             0
B_INDEX_ROWID              "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_INDEX_USER               "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_OUT_OF_ROW               0
B_STORAGE                  "ON PM_BDATA"
D_CLUSTER_ALL              0
D_CLUSTER_DELETED_AT       1
D_INDEX_ALL                "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_INDEX_DELETED_AT         "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_STORAGE                  "ON PM_DDATA"
END

##NETWORK_DEFAULTS::NETWORK
A_CLUSTER_ROWID            1
A_CLUSTER_STATEID          0
A_CLUSTER_USER             0
A_INDEX_ROWID              "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_STATEID            "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_USER               "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_OUT_OF_ROW               0
A_STORAGE                  "ON PM_ADATA"
B_CLUSTER_ROWID            1
B_CLUSTER_USER             0
B_INDEX_ROWID              "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_INDEX_USER               "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_OUT_OF_ROW               0
B_STORAGE                  "ON PM_BDATA"
D_CLUSTER_ALL              0
D_CLUSTER_DELETED_AT       1
D_INDEX_ALL                "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_INDEX_DELETED_AT         "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_STORAGE                  "ON PM_DDATA"
END

##NETWORK_GEOGRAPHY
GEOMETRY_STORAGE           "GEOGRAPHY"
UI_NETWORK_TEXT            "User Interface description for SQL Server GEOGRAPHY"
END

##NETWORK_GEOGRAPHY::DESC
GEOMETRY_STORAGE           "GEOGRAPHY"
END

##NETWORK_GEOGRAPHY::NETWORK
GEOMETRY_STORAGE           "GEOGRAPHY"
END

##NETWORK_GEOMETRY
GEOMETRY_STORAGE           "GEOMETRY"
UI_NETWORK_TEXT            "User Interface description for SQL Server GEOMETRY"
END
```

J10067

```
##NETWORK_GEOMETRY::DESC
GEOMETRY_STORAGE            "GEOMETRY"
END

##NETWORK_GEOMETRY::NETWORK
GEOMETRY_STORAGE            "GEOMETRY"
END

##NETWORK_SDEBINARY
GEOMETRY_STORAGE            "SDEBINARY"
UI_NETWORK_TEXT             "User Interface description for SQL Server SDEBINARY"
END

##NETWORK_SDEBINARY::DESC
GEOMETRY_STORAGE            "SDEBINARY"
END

##NETWORK_SDEBINARY::NETWORK
GEOMETRY_STORAGE            "SDEBINARY"
END

##SDEBINARY
GEOMETRY_STORAGE            "SDEBINARY"
UI_TEXT                     "User Interface text description for SDEBINARY"
END

##SURVEY_MULTI_BINARY
UI_TEXT                     ""
END

##TERRAIN_DEFAULTS
A_CLUSTER_ROWID             0
A_CLUSTER_SHAPE             1
A_CLUSTER_STATEID           0
A_CLUSTER_USER              0
A_INDEX_ROWID               "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_SHAPE               "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_STATEID             "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_USER                "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_OUT_OF_ROW                0
A_STORAGE                   "ON PM_ADATA"
B_CLUSTER_ROWID             0
B_CLUSTER_SHAPE             1
B_CLUSTER_USER              0
B_INDEX_ROWID               "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_INDEX_SHAPE               "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_INDEX_USER                "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_OUT_OF_ROW                0
B_STORAGE                   "ON PM_BDATA"
D_CLUSTER_ALL               0
D_CLUSTER_DELETED_AT        1
D_INDEX_ALL                 "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_INDEX_DELETED_AT          "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_STORAGE                   "ON PM_DDATA"
F_CLUSTER_FID               1
F_INDEX_AREA                "WITH FILLFACTOR = 75 ON PM_FINDEX"
F_INDEX_FID                 "WITH FILLFACTOR = 75 ON PM_FINDEX"
F_INDEX_LEN                 "WITH FILLFACTOR = 75 ON PM_FINDEX"
F_OUT_OF_ROW                0
F_STORAGE                   "ON PM_FDATA"
S_CLUSTER_ALL               1
S_CLUSTER_SP_FID            0
S_INDEX_ALL                 "WITH FILLFACTOR = 75 ON PM_SINDEX"
S_INDEX_SP_FID              "WITH FILLFACTOR = 75 ON PM_SINDEX"
S_STORAGE                   "ON PM_SDATA"
UI_TERRAIN_TEXT             "The terrain default configuration"
END

##TERRAIN_DEFAULTS::EMBEDDED
A_CLUSTER_ROWID             0
A_CLUSTER_SHAPE             1
A_CLUSTER_STATEID           0
A_CLUSTER_USER              0
A_INDEX_ROWID               "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_SHAPE               "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_STATEID             "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_USER                "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_OUT_OF_ROW                0
A_STORAGE                   "ON PM_ADATA"
B_CLUSTER_ROWID             0
B_CLUSTER_SHAPE             1
B_CLUSTER_USER              0
B_INDEX_ROWID               "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_INDEX_SHAPE               "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_INDEX_USER                "WITH FILLFACTOR = 75 ON PM_BINDEX"
B_OUT_OF_ROW                0
B_STORAGE                   "ON PM_BDATA"
D_CLUSTER_ALL               0
```

```
D_CLUSTER_DELETED_AT       1
D_INDEX_ALL                "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_INDEX_DELETED_AT         "WITH FILLFACTOR = 90 ON PM_DINDEX"
D_STORAGE                  "ON PM_DDATA"
F_CLUSTER_FID              1
F_INDEX_AREA               "WITH FILLFACTOR = 75 ON PM_FINDEX"
F_INDEX_FID                "WITH FILLFACTOR = 75 ON PM_FINDEX"
F_INDEX_LEN                "WITH FILLFACTOR = 75 ON PM_FINDEX"
F_OUT_OF_ROW               0
F_STORAGE                  "ON PM_FDATA"
S_CLUSTER_ALL              1
S_CLUSTER_SP_FID           0
S_INDEX_ALL                "WITH FILLFACTOR = 75 ON PM_SINDEX"
S_INDEX_SP_FID             "WITH FILLFACTOR = 75 ON PM_SINDEX"
S_STORAGE                  "ON PM_SDATA"
END

##TERRAIN_GEOGRAPHY
GEOMETRY_STORAGE           "GEOGRAPHY"
UI_TERRAIN_TEXT            "User Interface description for SQL Server GEOGRAPHY"
END

##TERRAIN_GEOGRAPHY::EMBEDDED
GEOMETRY_STORAGE           "GEOGRAPHY"
END

##TERRAIN_GEOMETRY
GEOMETRY_STORAGE           "GEOMETRY"
UI_TERRAIN_TEXT            "User Interface description for SQL Server GEOMETRY"
END

##TERRAIN_GEOMETRY::EMBEDDED
GEOMETRY_STORAGE           "GEOMETRY"
END

##TERRAIN_SDEBINARY
GEOMETRY_STORAGE           "SDEBINARY"
UI_TERRAIN_TEXT            "User Interface description for SQL Server SDEBINARY"
END

##TERRAIN_SDEBINARY::EMBEDDED
GEOMETRY_STORAGE           "SDEBINARY"
END

##TOPOLOGY_DEFAULTS
A_CLUSTER_ROWID            0
A_CLUSTER_SHAPE            1
A_CLUSTER_STATEID          0
A_CLUSTER_USER             0
A_INDEX_ROWID              "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_SHAPE              "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_STATEID            "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_USER               "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_OUT_OF_ROW               0
A_STORAGE                  "ON PM_ADATA"
B_CLUSTER_ROWID            0
B_CLUSTER_SHAPE            1
B_CLUSTER_USER             0
B_INDEX_ROWID              "WITH FILLFACTOR = 75 ON PM_BINDEX_TOPO"
B_INDEX_SHAPE              "WITH FILLFACTOR = 75 ON PM_BINDEX_TOPO"
B_INDEX_USER               "WITH FILLFACTOR = 75 ON PM_BINDEX_TOPO"
B_OUT_OF_ROW               0
B_STORAGE                  "ON PM_BDATA_TOPO"
D_CLUSTER_ALL              0
D_CLUSTER_DELETED_AT       1
D_INDEX_ALL                "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_INDEX_DELETED_AT         "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_STORAGE                  "ON PM_DDATA"
F_CLUSTER_FID              1
F_INDEX_AREA               "WITH FILLFACTOR = 75 ON PM_FINDEX_TOPO"
F_INDEX_FID                "WITH FILLFACTOR = 75 ON PM_FINDEX_TOPO"
F_INDEX_LEN                "WITH FILLFACTOR = 75 ON PM_FINDEX_TOPO"
F_OUT_OF_ROW               0
F_STORAGE                  "ON PM_FDATA_TOPO"
S_CLUSTER_ALL              1
S_CLUSTER_SP_FID           0
S_INDEX_ALL                "WITH FILLFACTOR = 75 ON PM_SINDEX_TOPO"
S_INDEX_SP_FID             "WITH FILLFACTOR = 75 ON PM_SINDEX_TOPO"
S_STORAGE                  "ON PM_SDATA_TOPO"
UI_TOPOLOGY_TEXT           "The topology default configuration"
END

##TOPOLOGY_DEFAULTS::DIRTYAREAS
A_CLUSTER_ROWID            0
A_CLUSTER_SHAPE            1
A_CLUSTER_STATEID          0
A_CLUSTER_USER             0
A_INDEX_ROWID              "WITH FILLFACTOR = 75 ON PM_AINDEX"
```

```
A_INDEX_SHAPE              "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_STATEID            "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_INDEX_USER               "WITH FILLFACTOR = 75 ON PM_AINDEX"
A_OUT_OF_ROW               0
A_STORAGE                  "ON PM_ADATA"
B_CLUSTER_ROWID            0
B_CLUSTER_SHAPE            1
B_CLUSTER_USER             0
B_INDEX_ROWID              "WITH FILLFACTOR = 75 ON PM_BINDEX_TOPO"
B_INDEX_SHAPE              "WITH FILLFACTOR = 75 ON PM_BINDEX_TOPO"
B_INDEX_USER               "WITH FILLFACTOR = 75 ON PM_BINDEX_TOPO"
B_OUT_OF_ROW               0
B_STORAGE                  "ON PM_BDATA_TOPO"
D_CLUSTER_ALL              0
D_CLUSTER_DELETED_AT       1
D_INDEX_ALL                "WITH FILLFACTOR = 75 ON PM_DINDEX"
D_INDEX_DELETED_AT         "WITH FILLFACTOR = 90 ON PM_DINDEX"
D_STORAGE                  "ON PM_DDATA"
F_CLUSTER_FID              1
F_INDEX_AREA               "WITH FILLFACTOR = 75 ON PM_FINDEX_TOPO"
F_INDEX_FID                "WITH FILLFACTOR = 75 ON PM_FINDEX_TOPO"
F_INDEX_LEN                "WITH FILLFACTOR = 75 ON PM_FINDEX_TOPO"
F_OUT_OF_ROW               0
F_STORAGE                  "ON PM_FDATA_TOPO"
S_CLUSTER_ALL              1
S_CLUSTER_SP_FID           0
S_INDEX_ALL                "WITH FILLFACTOR = 75 ON PM_SINDEX_TOPO"
S_INDEX_SP_FID             "WITH FILLFACTOR = 75 ON PM_SINDEX_TOPO"
S_STORAGE                  "ON PM_SDATA_TOPO"
END

##TOPOLOGY_GEOGRAPHY
GEOMETRY_STORAGE           "GEOGRAPHY"
UI_TOPOLOGY_TEXT           "User Interface description for SQL Server GEOGRAPHY"
END

##TOPOLOGY_GEOGRAPHY::DIRTYAREAS
GEOMETRY_STORAGE           "GEOGRAPHY"
END

##TOPOLOGY_GEOMETRY
GEOMETRY_STORAGE           "GEOMETRY"
UI_TOPOLOGY_TEXT           "User Interface description for SQL Server GEOMETRY"
END

##TOPOLOGY_GEOMETRY::DIRTYAREAS
GEOMETRY_STORAGE           "GEOMETRY"
END

##TOPOLOGY_SDEBINARY
GEOMETRY_STORAGE           "SDEBINARY"
UI_TOPOLOGY_TEXT           "User Interface description for SQL Server SDEBINARY"
END

##TOPOLOGY_SDEBINARY::DIRTYAREAS
GEOMETRY_STORAGE           "SDEBINARY"
END

##WKB_GEOMETRY
GEOMETRY_STORAGE           "OGCWKB"
UI_TEXT                    "User Interface text description for OGC WKB"
END
```

Import the modified ***dbtune_pm.sde*** file.

```
sdedbtune -o import -f dbtune_pm.sde -u sde -p sde -i sde:sqlserver:mcsdbsrv -s mcsdbsrv –D pmdb
```

Step 4: Configure SQL Server Parameters

It is recommended that you use the following parameter values when creating a SQL Server database:

SQL Server Parameters

Parameter Name	Value
Server Memory: Use AWE to allocate memory	Enabled
Processors: Boost SQL Server Priority	Enabled

Parameter Name	Value
Security SQL Server and Windows Authentication mode	Enabled
Connections: Maximum number of concurrent connections	0 = unlimited
Connections: Allow remote connections to this server	Enabled

Step 5: Configure ArcSDE Parameters

You must configure the MAXBLOBSIZE and TCPKEEPALIVE parameters for the ArcSDE geodatabase used as the product library. The MAXBLOBSIZE value is -1 by default. However, if you are using SQL Server or another enterprise DBMS, make sure that this value is set to -1 and the TCPKEEPALIVE value is set to 1. This command should be used at the command prompt of a machine where ArcSDE is installed.

```
sdeconfig -o alter -v MAXBLOBSIZE=-1-i <service> -u sde -p <sde_password>
sdeconfig -o alter -v TCPKEEPALIVE=1 -i <service> -u sde -p <sde_password>
```

For more information, see the ArcSDE Administration Command Reference.

Step 6: Load the Data

Prepare your geodatabase for data loading.

Back up your database.

Change the ArcSDE buffer size:

```
sdeconfig -o alter -i sde:sqlserver:mcsdbsrv -s mcsdbsrv –D pmdb -u sde -p sdeadmin  -N -q
-v MINBUFSIZE=409600
sdeconfig -o alter -i sde:sqlserver:mcsdbsrv -s mcsdbsrv –D pmdb -u sde -p sdeadmin -N -q
-v MAXBUFSIZE=819200
```

Set the ArcSDE temp folder:

```
sdeconfig -o alter -i sde:sqlserver:mcsdbsrv -s mcsdbsrv –D pmdb -u sde -p sdeadmin -N -q
-v TEMP=C:\TEMP
```

List the ArcSDE parameters:

```
sdeconfig -o list -i sde:sqlserver:mcsdbsrv -s mcsdbsrv –D pmdb -u sde -p sdeadmin
```

Set the database recovery model to simple:

```
ALTER DATABASE [pmdb] SET RECOVERY SIMPLE WITH NO_WAIT
```

Increase the log file size:

```
ALTER DATABASE [pmdb] MODIFY FILE ( NAME = N'pmdb_Log01', MAXSIZE = 20480000KB )  --20GB
```

Load 10 percent of the data, estimate the total size of each data file, then resize the data files accordingly.

If loading or appending data to an existing feature class, even if the feature class is empty but you have to load a large amount of data, change the layer I/O mode to load_only_io; loading will be faster because indexes are disabled.

```
sdelayer –o  load_only_io -l contour_l,shape –D pmdb -i sde:sqlserver:mcsdbsrv -s mcsdbsrv -s mysrv
-u sde -p sde
```

Load your data model with the PM user, then load the production mapping data.

Back up your database.

Step 7: Register as Versioned

In ArcCatalog™, register the PM schema as versioned.

Step 8: Verify Storage

Run the SQL queries below to verify that the production tables and indexes were created under the correct FileGroups:

```
USE PMDB
GO
```

List FileGroups and data files:

```
EXEC sp_helpdb pmdb
GO
```

List FileGroups data files:

```
EXEC sp_helpfilegroup 'PRIMARY'
GO
```

List tables by FileGroups:

```
SELECT USER_NAME(o.uid) [Owner],
 OBJECT_NAME(i.id) [Table Name],
 FILEGROUP_NAME(groupid) AS [Filegroup Name]
 FROM sysindexes i inner join sysobjects o
 ON i.id = o.id
 WHERE i.indid IN (0, 1) AND OBJECTPROPERTY(i.id, 'IsMSShipped') = 0 AND
 USER_NAME(o.uid) = 'pm'
 ORDER BY 1,3,2
GO
```

List indexes by table and FileGroups:

```
select 'owner'=user_name(o.uid)
,'table_name'=object_name(i.id),i.indid
,'index_name'=i.name ,i.groupid
,'filegroup'=f.name ,'file_name'=d.physical_name
,'dataspace'=s.name from sys.sysindexes i
,sys.sysobjects o,sys.filegroups f
,sys.database_files d, sys.data_spaces s
where objectproperty(i.id,'IsUserTable') = 1
and i.id = o.id
and f.data_space_id = i.groupid
and f.data_space_id = d.data_space_id
and f.data_space_id = s.data_space_id
and user_name(o.uid) = 'pm'
order by object_name(i.id),i.name,f.name
go
```

If any tables or indexes are stored in the wrong FileGroup, use ALTER TABLE and ALTER INDEX to change the FileGroup (see SQL Server Books Online at http://msdn.microsoft.com/en-us/library/ms130214.aspx).

Also, in Management Studio, you can re-create the DDL script of tables and indexes. Then, within *create script*, you can modify the FileGroup parameter and re-create the tables and indexes in the correct file groups. This is particularly useful when tables are empty and you are allowed to re-create database objects.

Step 9: Prepare Geodatabase for Editing

Prepare the geodatabase for normal online transaction processing (OLTP) editing.

After loading the data, change the layer to normal I/O:

```
sdelayer –o  normal_io -l contour_l,shape –D pmdb -i sde:sqlserver:mcsdbsrv -s mcsdbsrv
-u sde -p sde
```

Change the ArcSDE buffer size:

```
sdeconfig -o alter -i sde:sqlserver:mcsdbsrv -s mcsdbsrv –D pmdb -u sde -p sdeadmin  -N -q -v
MINBUFSIZE=16384
sdeconfig -o alter -i sde:sqlserver:mcsdbsrv -s mcsdbsrv –D pmdb -u sde -p sdeadmin -N -q -v
MAXBUFSIZE=65536
```

Shrink the log file:

```
DBCC SHRINKFILE (pmdb_Log01)
```

Shrink the database:

```
DBCC SHRINKDATABASE (pmdb)
```

Update the statistics:

```
USE pmdb
GO
EXEC sp_updatestats
GO
```

Set the database recovery model to full:

```
ALTER DATABASE [pmdb] SET RECOVERY FULL WITH NO_WAIT
```

Back up your database.

Step 10: Grant Permissions and Verify Roles

Grant permissions to the PM tables through the schema:

```
USE [pmdb]
GO
EXEC sp_addrole 'pm_editor', 'pm'
GO
GRANT DELETE ON SCHEMA::[pm] TO [pm_editor]
GO
GRANT EXECUTE ON SCHEMA::[pm] TO [pm_editor]
GO
GRANT INSERT ON SCHEMA::[pm] TO [pm_editor]
GO
GRANT SELECT ON SCHEMA::[pm] TO [pm_editor]
GO
GRANT UPDATE ON SCHEMA::[pm] TO [pm_editor]
GO
EXEC sp_addrole 'pm_viewer', 'pm'
GO
GRANT SELECT ON SCHEMA::[pm] TO [pm_viewer]
GO
```

Verify role permissions:

```
select USER_NAME(p.grantee_principal_id) AS principal_name,
 dp.type_desc AS principal_type_desc,
 p.class_desc,
 OBJECT_NAME(p.major_id) AS object_name,
 p.permission_name,
 p.state_desc AS permission_state_desc
from sys.database_permissions p
inner JOIN sys.database_principals dp
on p.grantee_principal_id = dp.principal_id
where USER_NAME(p.grantee_principal_id) in ('pm_editor','pm_viewer')
GO
```

Grant the PM_EDITOR role to ArcSDE editor users and the PM_VIEWER role to ArcSDE viewer users.

Step 11: Configure Log File Tables

Enterprise geodatabases use log file tables to maintain lists of selected records. Records are written to log file tables for later use by the application whenever a selection of a specific size is made, a reconciliation or post on a versioned database is performed, or a disconnected editing checkout is done in a client application. The log file tables store the ObjectIDs of the selected features so they can be redisplayed. This allows faster analysis and processing of information.

In ArcGIS, by default, log file tables are used if the selection set contains 100 or more records. This selection threshold of 100 features is set in the registry. It can be changed; however, Esri does not recommend doing so. There is no proven performance reason for changing it, and doing so could even cause performance problems. Thus, log file tables store feature selections in ArcMap that have more than 100 records for each connected ArcSDE editor/viewer user.

Log file options are set using specific parameters in the SERVER_CONFIG and DBTUNE tables (sde_server_config and sde_dbtune in a SQL Server database). Parameters in these tables are altered using the sdeconfig and sdedbtune commands, respectively.

In SQL Server, one table is created in tempdb in the format ##SDE_session<sde_id>. This table is truncated when the connecting application deletes its log files, and the table is dropped when the session disconnects. When using the default setting, users do not require CREATE TABLE permission in the database for the session table to be created in tempdb.

Learn more about ArcSDE log file tables at
Log file table configuration options for geodatabases in SQL Server.

Step 12: Create ArcSDE Users

The example below shows how to create an editor and viewer ArcSDE user.

Editor User

```
USE master
GO
CREATE LOGIN giseditor WITH PASSWORD = 'giseditor',
DEFAULT_DATABASE=[pmdb],DEFAULT_LANGUAGE=[us_english], CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
GO
```

Create user:

```
USE [pmdb]
GO
CREATE USER [giseditor] FOR LOGIN [giseditor]
GO
```

Grant privileges:

```
USE [pmdb]
GO
EXEC sp_addrolemember N'pm_editor', N'giseditor' GO
GO
```

Viewer User

```
USE master
GO
CREATE LOGIN gisviewer WITH PASSWORD = 'gisviewer',
DEFAULT_DATABASE=[pmdb],DEFAULT_LANGUAGE=[us_english], CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
GO
```

Create user:

```
USE [pmdb]
GO
CREATE USER [gisviewer] FOR LOGIN [gisviewer]
GO
```

Grant privileges:

```
USE [pmdb]
GO
EXEC sp_addrolemember N'pm_viewer', N'gisviewer' GO
GO
```

Conclusion

You can reduce disk contention and improve database I/O by storing the production mapping data in different locations on disk. However, this practice alone does not guarantee optimal database performance, and additional tuning tasks may be needed.

Learn more about the recommended tuning tasks:

Minimize disk I/O contention in SQL Server
What type of maintenance is needed for a geodatabase?

For more information on Esri Production Mapping, visit esri.com/productionmapping or e-mail productionmapping@esri.com.

Access blogs, forums, downloads, and more, from the Esri Production Mapping resource center.

You can access other resources at ArcGIS 10.1 for Desktop Help and Esri Support.

Esri inspires and enables people to positively impact their future through a deeper, geographic understanding of the changing world around them.

Governments, industry leaders, academics, and nongovernmental organizations trust us to connect them with the analytic knowledge they need to make the critical decisions that shape the planet. For more than 40 years, Esri has cultivated collaborative relationships with partners who share our commitment to solving earth's most pressing challenges with geographic expertise and rational resolve. Today, we believe that geography is at the heart of a more resilient and sustainable future. Creating responsible products and solutions drives our passion for improving quality of life everywhere.

Contact Esri

380 New York Street
Redlands, California 92373-8100 USA

1 800 447 9778
T 909 793 2853
F 909 793 5953
info@esri.com
esri.com

Offices worldwide
esri.com/locations

Printed in USA