An Esri® White Paper • April 2011

Best Practices for Storing the Product Library Workspace in an Enterprise Geodatabase for SQL Server

Esri, 380 New York St., Redlands, CA 92373-8100 USA
TEL 909-793-2853 • FAX 909-793-5953 • E-MAIL info@esri.com • WEB esri.com

Best Practices for Storing the Product Library Workspace in an Enterprise Geodatabase for SQL Server

An Esri White Paper

Contents **Page**

Contents **Page**

Best Practices for Storing the Product Library Workspace in an Enterprise Geodatabase for SQL Server

Introduction

The product library is part of Esri® Production Mapping. It is a geodatabase that allows multiuser environments to centralize information and behavior for cartographic and digital data production. Production business rules, documents, and spatial information are stored inside the product library, allowing an organization to enforce and standardize production. Data model information, data validation rules, geographic extents, symbology rules, and map documents can all be managed inside the product library as examples of production business rules. In other words, the product library is essentially a geographic document management system. When stored in an enterprise geodatabase, the workspace supports versioning. This white paper is intended for database administrators to help them establish the product library workspace in an enterprise geodatabase for Microsoft® SQL Server®. The enterprise geodatabase uses ArcSDE® technology as the gateway between geographic information system (GIS) clients and SQL Server.

Overview of SDE DBTUNE

DBTUNE storage parameters let you control how ArcSDE technology creates objects within a SQL Server database. You can determine such things as how to allocate space to a table or index, which FileGroup 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. The DBTUNE table, 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.

If a large number of database connections are accessing the same files in the same location on the disk, database performance will be slow, 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 the disk.

Thus, DBTUNE can be modified to store the Reviewer feature dataset and tables in separate data files in different locations on the disk. This will lead to reduced disk contention and improved 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 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.

With data spread across multiple disks, more spindles 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 0
- 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

Reduce Disk I/O Contention

As a rule, you should create database files 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 help eliminate 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 same drive that contains the operating system files.

- Put transaction log files on a drive separate from data files. This gives 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 lets SQL Server perform tempdb operations in parallel with database operations.

- 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, you can 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://msdn.microsoft.com/en-us/ms190764.aspx and *Microsoft Windows® 2000 Server Administrator's Companion* (Microsoft Press), Chapter 7 Planning Fault Tolerance and Avoidance, by Charlie Russel and Sharon Crawford, at http://technet.microsoft.com/pt-br/library/bb742464(en-us).aspx.

- Use partitioning on large tables. Partitioning lets you split a table across multiple FileGroups by using partitions; 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

Step 1: Create Data Files

Create new FileGroups to store the product library feature classes and tables.

FILEGROUP	*SDE_PARAMETER*
PRODLIB_BDATA	Business table
PRODLIB_BINDEX	Business table index
PRODLIB_FDATA	Feature table
PRODLIB_FINDEX	Feature table index
PRODLIB_SDATA	Spatial Index table
PRODLIB_SINDEX	Spatial Index table index
PRODLIB_ADATA	Adds table (versioned)
PRODLIB_AINDEX	Adds table index
PRODLIB_DDATA	Deletes table (versioned)
PRODLIB_DINDEX	Deletes table index

```
USE MASTER
GO
ALTER DATABASE [PRODLIB] ADD FILEGROUP [PRODLIB_BDATA]
GO
ALTER DATABASE [PRODLIB] ADD FILE(NAME = N'prodlib_Bdata01', FILENAME =
N'C:\mssql\data\prodlib\prodlib_Bdata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_BDATA]
GO
ALTER DATABASE [PRODLIB] ADD FILEGROUP [PRODLIB_BINDEX]
GO
ALTER DATABASE [PRODLIB] ADD FILE(NAME = N'prodlib_Bindex01', FILENAME =
N'C:\mssql\data\prodlib\prodlib_Bindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_BINDEX]
GO
ALTER DATABASE [PRODLIB] ADD FILEGROUP [PRODLIB_FDATA]
GO
ALTER DATABASE [PRODLIB] ADD FILE(NAME = N'prodlib_Fdata01', FILENAME =
N'C:\mssql\data\prodlib\prodlib_Fdata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_FDATA]
GO
ALTER DATABASE [PRODLIB] ADD FILEGROUP [PRODLIB_FINDEX]
GO
ALTER DATABASE [PRODLIB] ADD FILE(NAME = N'prodlib_Findex01', FILENAME =
N'C:\mssql\data\prodlib\prodlib_Findex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_FINDEX]
GO
ALTER DATABASE [PRODLIB] ADD FILEGROUP [PRODLIB_SDATA]
GO
ALTER DATABASE [PRODLIB] ADD FILE(NAME = N'prodlib_Sdata01', FILENAME =
N'C:\mssql\data\prodlib\prodlib_Sdata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_SDATA]
GO
ALTER DATABASE [PRODLIB] ADD FILEGROUP [PRODLIB_SINDEX]
GO
ALTER DATABASE [PRODLIB] ADD FILE(NAME = N'prodlib_Sindex01', FILENAME =
N'C:\mssql\data\prodlib\prodlib_Sindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_SINDEX]
GO
ALTER DATABASE [PRODLIB] ADD FILEGROUP [PRODLIB_ADATA]
GO
ALTER DATABASE [PRODLIB] ADD FILE(NAME = N'prodlib_Adata01', FILENAME =
N'C:\mssql\data\prodlib\prodlib_Adata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_ADATA]
GO
ALTER DATABASE [PRODLIB] ADD FILEGROUP [PRODLIB_AINDEX]
GO
ALTER DATABASE [PRODLIB] ADD FILE(NAME = N'prodlib_Aindex01', FILENAME =
N'C:\mssql\data\prodlib\prodlib_Aindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_AINDEX]
GO
ALTER DATABASE [PRODLIB] ADD FILEGROUP [PRODLIB_DDATA]
GO
ALTER DATABASE [PRODLIB] ADD FILE(NAME = N'prodlib_Ddata01', FILENAME =
N'C:\mssql\data\prodlib\prodlib_Ddata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_DDATA]
GO
ALTER DATABASE [PRODLIB] ADD FILEGROUP [PRODLIB_DINDEX]
GO
ALTER DATABASE [PRODLIB] ADD FILE(NAME = N'prodlib_Dindex01', FILENAME =
N'C:\mssql\data\prodlib\prodlib_Dindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_DINDEX]
GO
```

Verify FileGroups and data files:

```
EXEC sp_helpdb prodlib
GO
```

Step 2: Create the ProdLib User

Create a new database user to store the product library feature classes and tables; grant the appropriate permissions.

Create user and schema:

```
USE [prodlib]
GO
EXEC sp_addlogin N'prodlib', 'prodlib', @logindb, @loginlang
GO
CREATE USER [prodlib] FOR LOGIN [prodlib]
GO
CREATE SCHEMA [prodlib] AUTHORIZATION [prodlib]
GO
ALTER USER [prodlib] WITH DEFAULT_SCHEMA=[prodlib]
GO
```

Grant privileges:

```
USE [prodlib]
GO
EXEC sp_droprolemember 'gis_data_creator', 'prodlib'
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', 'prodlib'
GO
```

Verify roles:

```
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) = 'prodlib'
```

Associate Login prodlib with User prodlib:

```
USE [prodlib]
GO
EXEC sp_change_users_login 'update_one','prodlib','prodlib'
GO
EXEC sp_helpuser 'prodlib'
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 5151 –D prodlib
```

Copy ***dbtune_exp.sde*** to ***dbtune_prodlib.sde***.

- Modify the ##DEFAULTS configuration keywords.

```
dbtune_prodlib.sde

##DEFAULTS
A_INDEX_RASTER          "WITH FILLFACTOR = 75 ON PRODLIB_AINDEX"
A_INDEX_ROWID           "WITH FILLFACTOR = 75 ON PRODLIB_AINDEX"
A_INDEX_SHAPE           "WITH FILLFACTOR = 75 ON PRODLIB_AINDEX"
A_INDEX_STATEID         "WITH FILLFACTOR = 75 ON PRODLIB_AINDEX"
A_INDEX_USER            "WITH FILLFACTOR = 75 ON PRODLIB_AINDEX"
A_INDEX_XML             "WITH FILLFACTOR = 75 ON PRODLIB_AINDEX"
A_STORAGE               "ON PRODLIB_ADATA"
AUX_INDEX_COMPOSITE     "WITH FILLFACTOR = 75 ON GIS_RASTERIDX"
AUX_STORAGE             "ON GIS_RASTER"
B_INDEX_RASTER          "WITH FILLFACTOR = 75 ON PRODLIB_BINDEX"
B_INDEX_ROWID           "WITH FILLFACTOR = 75 ON PRODLIB_BINDEX"
B_INDEX_SHAPE           "WITH FILLFACTOR = 75 ON PRODLIB_BINDEX"
B_INDEX_TO_DATE         "WITH FILLFACTOR = 75 ON PRODLIB_BINDEX"
B_INDEX_USER            "WITH FILLFACTOR = 75 ON PRODLIB_BINDEX"
B_INDEX_XML             "WITH FILLFACTOR = 75 ON PRODLIB_BINDEX"
B_STORAGE               "ON PRODLIB_BDATA"
BLK_INDEX_COMPOSITE     "WITH FILLFACTOR = 75 ON GIS_RASTERIDX"
BLK_STORAGE             "ON GIS_RASTER"
BND_INDEX_COMPOSITE     "WITH FILLFACTOR = 75 ON GIS_RASTERIDX"
BND_INDEX_ID            "WITH FILLFACTOR = 75 ON GIS_RASTERIDX"
BND_STORAGE             "ON GIS_RASTER"
D_INDEX_ALL             "WITH FILLFACTOR = 75 ON PRODLIB_DINDEX"
D_INDEX_DELETED_AT      "WITH FILLFACTOR = 75 ON PRODLIB_DINDEX"
D_STORAGE               "ON PRODLIB_DDATA"
F_INDEX_AREA            "WITH FILLFACTOR = 75 ON PRODLIB_FINDEX"
F_INDEX_FID             "WITH FILLFACTOR = 75 ON PRODLIB_FINDEX"
F_INDEX_LEN             "WITH FILLFACTOR = 75 ON PRODLIB_FINDEX"
F_STORAGE               "ON PRODLIB_FDATA"
GEOMETRY_STORAGE        "SDEBINARY"
GEOMTAB_PK              "WITH FILLFACTOR = 75"
RAS_INDEX_ID            "WITH FILLFACTOR = 75 ON GIS_RASTERIDX"
RAS_STORAGE             "ON GIS_RASTER"
S_INDEX_ALL             "WITH FILLFACTOR = 75 ON PRODLIB_SINDEX"
S_INDEX_SP_FID          "WITH FILLFACTOR = 75 ON PRODLIB_SINDEX"
S_STORAGE               "ON PRODLIB_SDATA"
END
```

- Import the modified ***dbtune_prodlib.sde*** file.

```
sdedbtune -o import -f dbtune_prodlib.sde -u sde -p sde -i 5151 –D prodlib
```

Step 4: Create the Product Library Database Connection

Create a database connection in ArcCatalog™ with the PRODLIB user; this will be the product library workspace location.

Step 5: Configure SQL Server Parameters

It is recommended that you ensure that the following parameter values are used when creating a SQL Server database.

Parameter name	Value
Server Memory: Use AWE to allocate memory	Enabled
Processors: Boost SQL Server Priority	Enabled
Security SQL Server and Windows Authentication mode	Enabled
Connections: Maximum number of concurrent connections	0 = unlimited
Connections: Allow remote connections to this server	Enabled

SQL Server Parameters for Product Library

Step 6: Configure ArcSDE Parameters

You need to 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 from the command prompt of a machine where ArcSDE is installed.

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

For more information, see the ArcSDE Administration Command Reference.

Step 7: Create the CKB_USERS Role

A role needs to be assigned to the users that are going to be working with the product library so they can view or add components, information, and data. The role CKB_USERS must be created for the users to be recognized by the product library. This can be done by using the following statement:

```
USE [prodlib]
GO
EXEC sp_droprole 'ckb_users'
GO
EXEC sp_addrole 'ckb_users', 'prodlib'
GO
```

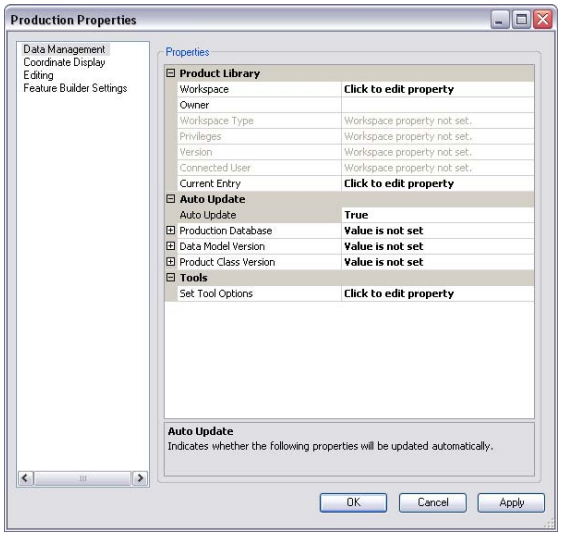
Step 8: Create the Product Library Workspace

After the geodatabase has been created, various tables and feature classes that are part of the product library need to be added to it. This process can be completed in ArcMap.

The steps in this section are for defining and upgrading the geodatabase as a product library in ArcMap.

1. Open ArcMap.
2. On the main menu, click **Customize > Production > Production Properties**.

   The **Production Properties** dialog box appears.
3. If necessary, click **Data Management**.



4. Click the cell next to **Workspace** in the **Product Library** section and click the ellipsis (...) that appears.

   The **Please browse to the location of the Product Library Workspace** dialog box appears.

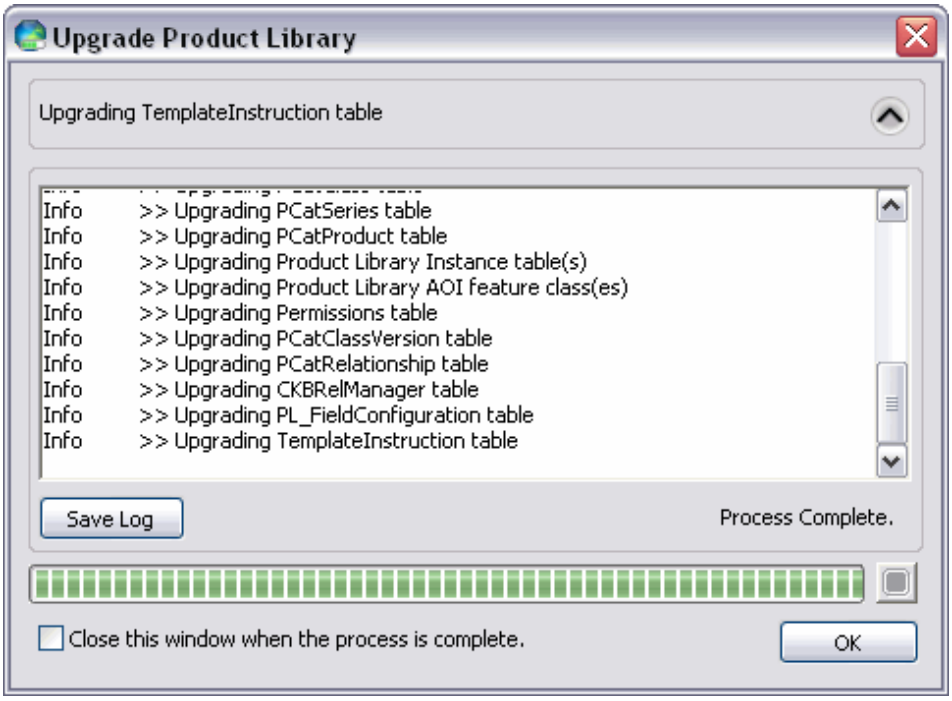
5. Navigate to the Product Library database.

6. Click **Open**.

   The **Upgrade Workspace** dialog box appears.

7. Click **OK** to upgrade the geodatabase you want to use as the **product library**.

   The **Upgrade Product Library** dialog box appears with the progress of the upgrade.



8. Click **OK** when the process completes.

   The **Production Properties** dialog box appears.

9. Click **OK**.

Step 9: Verify the Storage

Run the SQL queries below to verify if the product library workspace was created under the correct FileGroups.

```
USE [prodlib]
GO
```

List FileGroups and data files:

```
EXEC sp_helpdb prodlib
GO
```

List FileGroup data files:

```
USE [prodlib]
GO
EXEC sp_helpfilegroup 'PRIMARY'
GO
```

List tables by FileGroup:

```
USE [prodlib]
GO
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) = 'prodlib'
 ORDER BY 1,3,2
GO
```

List indexes by table and FileGroup:

```
USE [prodlib]
GO
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) = 'prodlib'
order by object_name(i.id),i.name,f.name
GO
```

If any tables or indexes are stored in the wrong FileGroup, then ALTER TABLE and ALTER INDEX can be used 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 FileGroups. This is particularly useful when tables are empty and you are allowed to re-create database objects.

Step 10: Register as Versioned

You need to register product library components for versioning to allow them to be editable.

1. Open ArcCatalog.
2. Expand Database Connections.
3. Double-click the product library administrator connection geodatabase to connect to it.
4. Right-click each feature class in your product library and choose **Register as Versioned**. Do not check the **Register the selected objects with the option to move edits to base** check box.
5. Click **OK**.
6. Right-click each table in your product library and choose **Register as Versioned**. Do not check the **Register the selected objects with the option to move edits to base** check box.
7. Click **OK**.

Step 11: Validate Permissions and Roles

All the tables in the product library need to have read/write privileges assigned to them except the PCAT_PERMISSION table. The PCAT_PERMISSION table only needs read privileges assigned to it. The permissions need to be assigned to the CKB_USERS role. You can re-create the CKB_USERS role and grant the right permissions to the role by using the following script:

```
/*RECREATE ckb_users role */
USE [prodlib]
GO
EXEC sp_droprolemember 'ckb_users', 'prodlibuser'
GO
EXEC sp_droprole 'ckb_users'
GO
EXEC sp_addrole 'ckb_users', 'prodlib'
GO
EXEC sp_addrolemember 'ckb_users', 'prodlibuser'
GO
DECLARE @OWNER varchar(10)
SET @OWNER = 'PRODLIB'
DECLARE Tables_Cursor CURSOR
READ_ONLY
FOR SELECT a.name as table_name, a.xtype as type
    FROM dbo.sysobjects a, dbo.sysusers b
    WHERE a.uid = b.uid and a.xtype in ('U','P') and b.name = @OWNER ORDER BY a.name
DECLARE @name varchar(100), @type varchar(1)
OPEN Tables_Cursor
FETCH NEXT FROM Tables_Cursor INTO @name, @type
WHILE (@@fetch_status <> -1)
BEGIN
  IF (@@fetch_status <> -2)
   BEGIN
      --PRINT @owner + '.' + @name + ' ' + @type
      -- GRANT PERMISSIONS TO TABLE
    IF @type = 'U'
      BEGIN
      --EXECUTE ('GRANT SELECT ON ' + @OWNER + '.' + @name + ' TO pl_user')
      EXECUTE ('GRANT SELECT, INSERT, UPDATE, DELETE ON ' + @OWNER + '.' + @name + ' TO ckb_users')
    END
      ELSE
        --GRANT PERMISSION TO STORE PROCEDURE
```

```
        IF @type = 'P'
        BEGIN
          EXECUTE ('GRANT EXEC ON ' + @OWNER + '.' + @name + ' TO ckb_users')
        END
  END
  FETCH NEXT FROM Tables_Cursor INTO @name, @type
END
CLOSE Tables_Cursor
DEALLOCATE Tables_Cursor
GO
REVOKE INSERT, UPDATE, DELETE ON GIS.PCAT_PERMISSION FROM ckb_users;
GO
```

Grant Permissions Using ArcCatalog

Both the administrator and other user accounts in the underlying database management system should have appropriate privileges and roles assigned to them. When you set up your connection to your spatial database, ensure that you are connecting as the appropriate user.

1. Open ArcCatalog.
2. Expand Database Connections.
3. Double-click the product library administrator connection geodatabase to connect to it.
4. Select all tables except PCAT_PERMISSION, right-click, then click **Privileges**.
5. Type CKB_USERS into the text box on the **Privileges** dialog box.
6. Check the check boxes next to **SELECT**, **UPDATE**, **INSERT**, and **DELETE**.
7. Click **OK**.
8. Select the PCAT_PERMISSION table, right-click, then click **Privileges**.
9. Type CKB_USERS into the text box on the **Privileges** dialog box.
10. Check the check box next to **SELECT**.
11. Click **OK**.

Step 12: 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 cause performance problems. Thus, log file tables store feature selections in ArcMap™ that are greater than 100 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.

The DBTUNE SESSION_TEMP_TABLE parameter must be set to 1 (true) to allow the session-based log file table to be created in tempdb. 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.

Learn more about ArcSDE log file tables at help.arcgis.com/en/arcgisdesktop/10.0/help/index.html#/Log_file_table_configuration_options_for_geodatabases_in_SQL_Server/002q00000012000000/.

Step 13: Create the SDE Product Library User

The example below shows how to create an ArcSDE user to access the product library:

```
USE master
GO
EXEC sp_addlogin N'prodlibuser', 'prodlibuser', @logindb, @loginlang
GO
```

Create user:

```
USE [prodlib]
GO
CREATE USER [prodlibuser] FOR LOGIN [prodlibuser]
GO
```

Grant privileges:

```
USE [prodlib]
GO
EXEC sp_addrolemember N'ckb_users', N'prodlibusers'
GO
```

Verify user permissions:

```
USE [prodlib]
GO
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) = 'prodlibuser'
GO
```

Step 14: Database Connections for Product Library Users

Database connections need to be created for the other product library users if the product library is stored in SQL Server. Create a database connection in ArcCatalog with the PRODLIBUSER user; this will be the product library sde connection. *Note:* You can create a login for each user in the product library database and, if using database authentication, type the user name of the product library user for whom you want to create a database connection.

Step 15: Product Library Permissions

There are two different levels of interaction with the product library in an SDE implementation of the product library: administrators and users. These levels of access are controlled through ArcCatalog database connections. The administrators manage the overall product library including the structure, components, and user permissions. This level of permissions through ArcCatalog database connections is related to the database role CKB_USERS.

Users can have varying degrees of access to parts of the product library based on whether they have edit, read/write, or read-only permissions based on their Windows login. Using the administrator's database connection, user accounts are created for anyone who is going to have access to the product library. To create a new user, first add the person as a product library user, then assign permissions.

Learn more about product library permissions at help.arcgis.com/en/arcgisdesktop/10.0/help/index.html#/Product_library_permissions/010300000043000000/.

Step 16: Add New Users to the Product Library

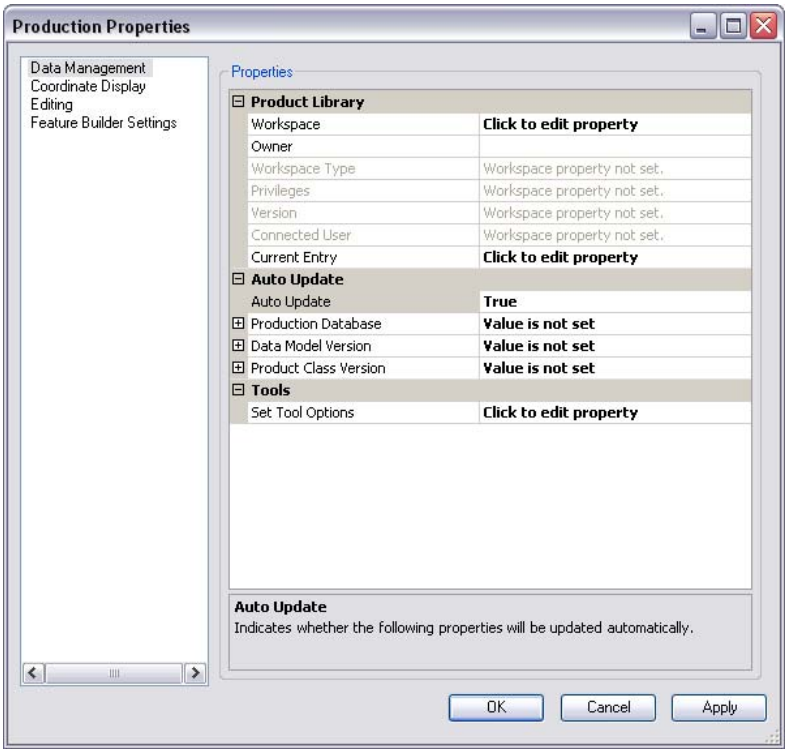
Using the SDE administrator's database connection, user accounts are created for anyone who is going to have access to the product library.

Initially, the user must be added to the geodatabase by the administrator. Each user is added using the first name, last name, and Windows user name.

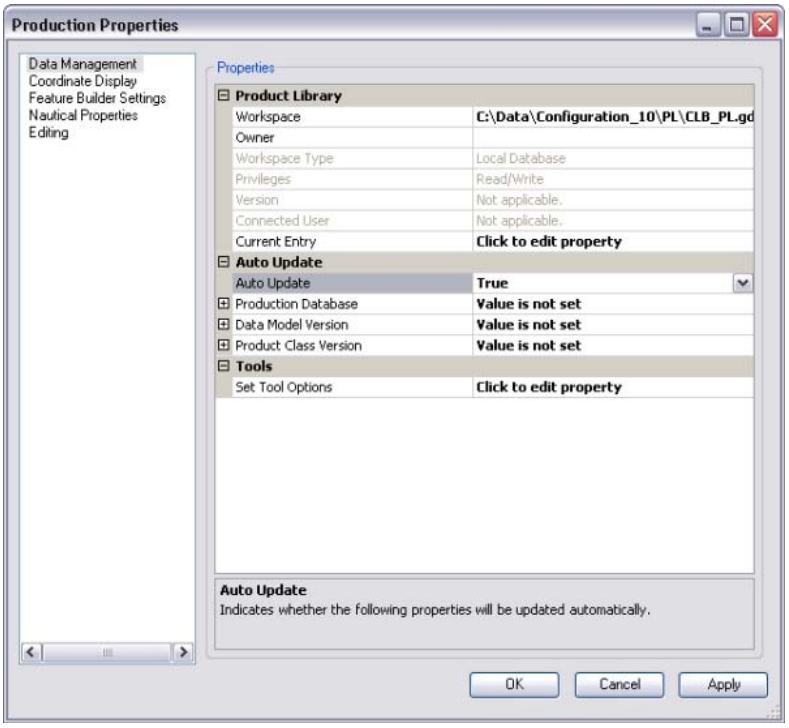
Steps:

1. Open ArcMap.
2. On the main menu, click **Customize** > **Production** > **Production Properties**.

   The **Production Properties** dialog box appears.
3. If necessary, click **Data Management**.

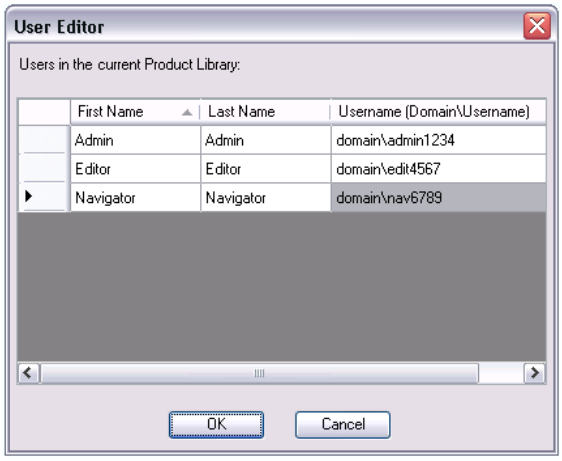


4. Click the cell next to **Set Tool Options**.



5. Click the drop-down arrow, then click the ellipsis (...) next to **Product Library Users**.

   The **User Editor** dialog box appears.



> **Tip:** If you are using an ArcSDE geodatabase as your product library, you can also add new users by right-clicking a series and clicking Permissions.

6. Right-click anywhere in the **Users in the current Product Library** list and choose **New User**.

   A new row appears in the list.

> **Tip:** If you are using an ArcSDE geodatabase as your product library, you can also add users when you are assigning permissions to existing users.

7. Type the user's first name in the **First Name** cell.
8. Type the user's last name in the **Last Name** cell.
9. Type the user's Windows login name in the **Users** cell.
10. Repeat steps 6–9 for each user you need to add to the product library.
11. Click **OK**.

    The **Data Management** pane appears.

12. Click **OK**.

Assigning Permissions to Users

Once the user is added, the permissions can be granted at the series level of the product library. Permissions are passed down to all products within a given series. Permissions are also passed up from the series to the class and the solution. For example, if a user is given permission to one or more series below a particular class or solution, the user has access to those entries. By default, the permissions for a user are set to Not Available, but there are four different levels:

- Not Available—The series and all components beneath it are hidden from the user.
- Read Only—Properties can be viewed for all levels of the product library.
- Check In/Check Out—Files can be checked in and out.
- Edit—Product library levels can be added, modified, and removed, and files can be checked in and out.

⚠ Caution: This only applies if you are using an ArcSDE geodatabase as your product library.

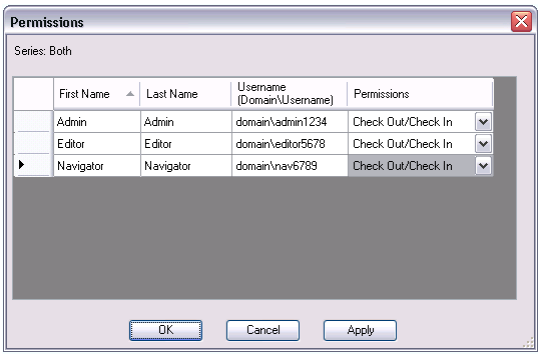
1. Open ArcMap.
2. On the menu bar, click **Customize > Production > Product Library**.

Tip: If **Product Library** is not enabled, you may need to enable the Production Mapping extension by clicking **Customize > Extensions** and checking the check box for **Production Mapping**.

Tip: You can also open the window by clicking the **Product Library Window** button on the **Production Cartography toolbar**.

3. Navigate to the series level of the product class for which you want to assign permissions.
4. Right-click the series name and click **Permissions**.

   The **Permissions** dialog box appears.



J10022

5. Click the **Permissions** drop-down arrow for the user to whom you want to grant permissions and choose an option.

6. Repeat step 5 for all users to whom you want to give permissions.

7. Click **OK**.

Conclusion

You can reduce disk contention and improve database I/O by storing the product library workspace in different locations on the 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 at help.arcgis.com/en/arcgisdesktop/10.0/help/index.html#/Minimize_disk_I_O_contention_in_SQL_Server/002q00000021000000/.

For more information on the product library, visit the Esri Production Mapping page at esri.com/software/arcgis/extensions/production-mapping/index.html.

Learn about setting up the product library in an ArcSDE environment at help.arcgis.com/en/arcgisdesktop/10.0/help/index.html#/Setting_up_the_product_library_in_an_ArcSDE_environment/010300000299000000/.

Access blogs, forums, downloads, and more, via the Esri Production Mapping Resource Center at resources.arcgis.com/content/esri-production-mapping/10.0/about.

You can access other resources at ArcGIS Desktop 10 Help at help.arcgis.com/en/arcgisdesktop/10.0/help/index.html and Esri Support at http://support.esri.com.

About Esri

Since 1969, Esri has been helping organizations map and model our world. Esri's GIS software tools and methodologies enable these organizations to effectively analyze and manage their geographic information and make better decisions. They are supported by our experienced and knowledgeable staff and extensive network of business partners and international distributors.

A full-service GIS company, Esri supports the implementation of GIS technology on desktops, servers, online services, and mobile devices. These GIS solutions are flexible, customizable, and easy to use.

Our Focus

Esri software is used by hundreds of thousands of organizations that apply GIS to solve problems and make our world a better place to live. We pay close attention to our users to ensure they have the best tools possible to accomplish their missions. A comprehensive suite of training options offered worldwide helps our users fully leverage their GIS applications.

Esri is a socially conscious business, actively supporting organizations involved in education, conservation, sustainable development, and humanitarian affairs.

Contact Esri

1 800 GIS XPRT (1 800 447 9778)
T 909 793 2853
F 909 793 5953
info@esri.com
esri.com

Offices worldwide
esri.com/locations

esri®
380 New York Street
Redlands, California 92373-8100 USA

Printed in USA