

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

An Esri® White Paper  
October 2012



Copyright © 2012 Esri  
All rights reserved.  
Printed in the United States of America.

The information contained in this document is the exclusive property of Esri. This work is protected under United States copyright law and other international copyright treaties and conventions. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, or by any information storage or retrieval system, except as expressly permitted in writing by Esri. All requests should be sent to Attention: Contracts and Legal Services Manager, Esri, 380 New York Street, Redlands, CA 92373-8100 USA.

The information contained in this document is subject to change without notice.

Esri, the Esri globe logo, ArcSDE, ArcCatalog, ArcMap, ArcGIS, [esri.com](http://esri.com), and [@esri.com](mailto:@esri.com) are trademarks, service marks, or registered marks of Esri in the United States, the European Community, or certain other jurisdictions. Other companies and products or services mentioned herein may be trademarks, service marks, or registered marks of their respective mark owners.

---

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

## An Esri White Paper

Contents	Page
Introduction.....	1
Overview of ArcSDE DBTUNE.....	1
Disk Configuration.....	2
Reducing Disk I/O Contention.....	2
Transparent Data Encryption .....	3
Step 1: Create Data Files.....	5
Step 2: Create the Prodlib User.....	7
Step 3: Modify DBTUNE .....	8
Step 4: Create the Product Library Database Connection .....	8
Step 5: Configure SQL Server Parameters .....	9
Step 6: Configure ArcSDE Parameters.....	9
Step 7: Create the CKB_USERS Role.....	9
Step 8: Create the Product Library Workspace.....	9
Step 9: Verify the Storage.....	11
Step 10: Register as Versioned .....	12

<b>Contents</b>	<b>Page</b>
Step 11: Validate Permissions and Roles.....	12
Grant Permissions Using ArcCatalog .....	13
Step 12: Configure Log File Tables.....	14
Step 13: Create the ArcSDE Product Library User .....	14
Step 14: Create Database Connections for Product Library Users .....	15
Step 15: Assign Product Library Permissions .....	15
Step 16: Add New Users to the Product Library .....	15
Assigning Permissions to Users.....	17
Replication .....	19
Conclusion .....	19

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

**Introduction** The product library is part of the Esri® Production Mapping extension. 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 to help database administrators establish the product library workspace in an enterprise geodatabase for SQL Server. The enterprise geodatabase uses ArcSDE® technology as the gateway between geographic information system (GIS) clients and SQL Server.

**Overview of ArcSDE 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.

When a large number of database connections access the same files in the same location on the 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 the disk.

Thus, DBTUNE can be modified to store the product library tables in separate data files in different locations on the 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 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 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 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 the 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 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 the transaction log files in a RAID 10 system and storing the 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](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/pt-br/library/bb742464\(en-us\).aspx](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

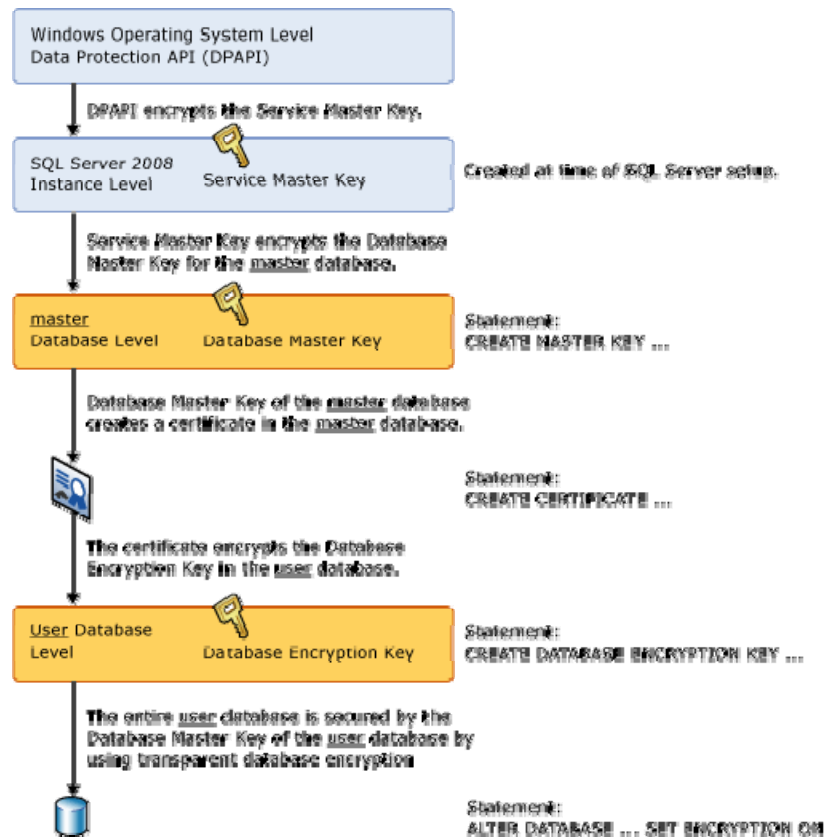
## Transparent Data Encryption

The precautions you can take to help secure 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 can just restore or attach the database and browse the data. One solution is to encrypt the 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 in an encrypted database 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.



J10022

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

/* Create Certificate */
CREATE CERTIFICATE prodlib_cert WITH SUBJECT = 'PRODLIB Server Certificate';
GO

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

/* Backup certificate */
BACKUP CERTIFICATE prodlib_cert TO FILE = 'D:\mssqlbackup\master\prodlib.cert'
    WITH PRIVATE KEY (
        FILE = 'D:\mssqlbackup\master\prodlib.pvk',
        ENCRYPTION BY PASSWORD = 'prodlibadmin');
GO

USE prodlibdb
GO
/* Create Encryption key */
CREATE DATABASE ENCRYPTION KEY
    WITH ALGORITHM = AES 256
    ENCRYPTION BY SERVER CERTIFICATE prodlib_cert;
GO

/* Encrypt database */
ALTER DATABASE prodlibdb 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 = 'prodlibdb'
GO
```

## Step 1: Create Data Files

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

FILEGROUP	ArcSDE_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 [PRODLIBDB] ADD FILEGROUP [PRODLIB_BDATA]
GO
ALTER DATABASE [PRODLIBDB] ADD FILE(NAME = N'prodlib_Bdata01', FILENAME =
N'C:\mssql\data\prodlibdb\prodlib_Bdata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_BDATA]
GO
ALTER DATABASE [PRODLIBDB] ADD FILEGROUP [PRODLIB_BINDEXT]
GO
ALTER DATABASE [PRODLIBDB] ADD FILE(NAME = N'prodlib_Bindex01', FILENAME =
N'C:\mssql\data\prodlibdb\prodlib_Bindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_BINDEXT]
GO
ALTER DATABASE [PRODLIBDB] ADD FILEGROUP [PRODLIB_FDATA]
GO
ALTER DATABASE [PRODLIBDB] ADD FILE(NAME = N'prodlib_Fdata01', FILENAME =
N'C:\mssql\data\prodlibdb\prodlib_Fdata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_FDATA]
GO
ALTER DATABASE [PRODLIBDB] ADD FILEGROUP [PRODLIB_FINDEX]
GO
ALTER DATABASE [PRODLIBDB] ADD FILE(NAME = N'prodlib_Findex01', FILENAME =
N'C:\mssql\data\prodlibdb\prodlib_Findex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_FINDEX]
GO
ALTER DATABASE [PRODLIBDB] ADD FILEGROUP [PRODLIB_SDATA]
GO
ALTER DATABASE [PRODLIBDB] ADD FILE(NAME = N'prodlib_Sdata01', FILENAME =
N'C:\mssql\data\prodlibdb\prodlib_Sdata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_SDATA]
GO
ALTER DATABASE [PRODLIBDB] ADD FILEGROUP [PRODLIB_SINDEX]
GO
ALTER DATABASE [PRODLIBDB] ADD FILE(NAME = N'prodlib_Sindex01', FILENAME =
N'C:\mssql\data\prodlibdb\prodlib_Sindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_SINDEX]
GO
ALTER DATABASE [PRODLIBDB] ADD FILEGROUP [PRODLIB_ADATA]
GO
ALTER DATABASE [PRODLIBDB] ADD FILE(NAME = N'prodlib_Adata01', FILENAME =
N'C:\mssql\data\prodlibdb\prodlib_Adata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_ADATA]
GO
ALTER DATABASE [PRODLIBDB] ADD FILEGROUP [PRODLIB_AINDEX]
GO
ALTER DATABASE [PRODLIBDB] ADD FILE(NAME = N'prodlib_Aindex01', FILENAME =
N'C:\mssql\data\prodlibdb\prodlib_Aindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_AINDEX]
GO
ALTER DATABASE [PRODLIBDB] ADD FILEGROUP [PRODLIB_DDATA]
GO
ALTER DATABASE [PRODLIBDB] ADD FILE(NAME = N'prodlib_Ddata01', FILENAME =
N'C:\mssql\data\prodlibdb\prodlib_Ddata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_DDATA]
GO
ALTER DATABASE [PRODLIBDB] ADD FILEGROUP [PRODLIB_DINDEX]
GO
ALTER DATABASE [PRODLIBDB] ADD FILE(NAME = N'prodlib_Dindex01', FILENAME =
N'C:\mssql\data\prodlibdb\prodlib_Dindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO
FILEGROUP [PRODLIB_DINDEX]
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 [PRODLIBDB] MODIFY FILE ( NAME = N'prodlib_Bdata01', SIZE = 400MB )
ALTER DATABASE [PRODLIBDB] MODIFY FILE ( NAME = N'prodlib_Bindex01', SIZE = 400MB )
ALTER DATABASE [PRODLIBDB] MODIFY FILE ( NAME = N'prodlib_Fdata01', SIZE = 400MB )
ALTER DATABASE [PRODLIBDB] MODIFY FILE ( NAME = N'prodlib_Findex01', SIZE = 400MB )
ALTER DATABASE [PRODLIBDB] MODIFY FILE ( NAME = N'prodlib_Sdata01', SIZE = 400MB )
ALTER DATABASE [PRODLIBDB] MODIFY FILE ( NAME = N'prodlib_Sindex01', SIZE = 400MB )
ALTER DATABASE [PRODLIBDB] MODIFY FILE ( NAME = N'prodlib_Adata01', SIZE = 400MB )
ALTER DATABASE [PRODLIBDB] MODIFY FILE ( NAME = N'prodlib_Aindex01', SIZE = 400MB )
ALTER DATABASE [PRODLIBDB] MODIFY FILE ( NAME = N'prodlib_Ddata01', SIZE = 400MB )
ALTER DATABASE [PRODLIBDB] MODIFY FILE ( NAME = N'prodlib_Dindex01', SIZE = 400MB )
```

### Verify FileGroups and data files:

```
EXEC sp_helpdb prodlibdb
GO
```

J10022

## 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 [prodlibdb]
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 [prodlibdb]
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 [prodlibdb]
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 sde:sqlserver:mcsdbsrv1 -D prodlibdb
```

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"
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"
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 "GEOMETRY"
GEOMTAB_PK "WITH FILLFACTOR = 75 ON PRODLIB_FINDEX"
GEOMTAB_STORAGE "ON PRODLIB_FDATA"
I STORAGE "ON PRODLIB_FDATA"
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
```

If your database only stores the Product Library repository, you can edit `##DEFAULTS`; otherwise, create a new configuration keyword as described above.

- Import the modified *dbtune\_prodlib.sde* file.

```
sdedbtune -o import -f dbtune_prodlib.sde -u sde -p sde -i sde:sqlserver:mcsdbsrv1 -D prodlibdb
```

## 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.

J10022

## Step 5: Configure SQL Server Parameters

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

### SQL Server Parameters for Product Library

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

## 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>
sdeconfig -o alter -v TCPKEEPALIVE=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 who 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 [prodlibdb]
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.

### Steps

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

### Tips:

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

- You can also open the window by clicking the **Product Library** button on the **Production Cartography** or **Production Editing** toolbar.

3. Right-click **Product Library** and click **Select Product Library**.

The **Choose Product Library Workspace** dialog box appears.

4. Navigate to the product library database.

5. Click **Open**.

The **Upgrade Product Library Workspace** or the **Create Product Library Workspace** dialog box appears if the geodatabase does not have all the components necessary to perform as a product library. Continue to step 6 if one of these dialog boxes appears.

**Note:** If you also have the Esri Nautical Solution installed, tables are added to the product library schema that are necessary for nautical production but do not impact nonnautical production.

If this is not a new product library, and if an error appears when you choose a geodatabase as the product library, it may need to be compacted or compressed to restore its integrity. Compacting applies to personal and file geodatabases, and compressing applies to enterprise geodatabases.

If you select an enterprise workspace, the **Select Product Library Owner** dialog box appears, and you can continue to step 7.

6. Perform the following steps to upgrade or create the geodatabase you want to use as the product library:

- a. If necessary, click the drop-down arrow in the **Select Configuration Keyword** area and choose an option.

Options vary based on whether the database is a personal, file, or ArcSDE geodatabase.

- b. Click **OK**.

- c. Click **OK** once the upgrade or creation completes.

7. If the product library is an enterprise geodatabase, click the owner and click **OK**.

The **Upgrade Product Library Workspace** or the **Create Product Library Workspace** dialog box appears if the geodatabase does not have all the components necessary to perform as a product library. Perform the following steps if you are an administrator and want to upgrade or create the geodatabase to use as the product library.

J10022

- a. If necessary, click the drop-down arrow in the Select Configuration Keyword area and choose an option.

Options vary based on whether the database is a personal, file, or ArcSDE geodatabase.

- b. Click **OK**.

- c. If necessary, click **OK** once the upgrade or creation completes.

## Step 9: Verify the Storage

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

```
USE [prodlibdb]
GO
```

### List FileGroups and data files:

```
EXEC sp_helpdb prodlibdb
GO
```

### List FileGroup data files:

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

### List tables by FileGroup:

```
USE [prodlibdb]
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 [prodlibdb]
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, 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 FileGroups. This is particularly useful when tables are empty and you are allowed to re-create database objects.

## Step 10: Register as Versioned

If you manually import the product library tables and feature classes, such as importing an XML file, you need to verify that all the tables and feature classes are registered as versioned. This allows the software to edit the tables as you create and work with your product library. However, you shouldn't create new versions of the product library tables; this can result in inconsistencies within the versions.

### Steps

1. Expand **Database Connections** in the **Catalog Tree** window.
2. Double-click the product library administrator connection geodatabase to connect to it.
3. Right-click each feature class in your product library and click **Manage > Register as Versioned**.

Do not check the **Register the selected objects with the option to move edits to base** check box.

4. Click **OK**.
5. Right-click each table in your product library, except for those listed below, and click **Manage > Register as Versioned**.

Do not check the **Register the selected objects with the option to move edits to base** check box.

Do not register the following tables for versioning:

- ELM\_CATEGORIES
- ELM\_ELEMENTS
- ELM\_PRODUCTS
- ELM\_SOLUTIONS

6. 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, which 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 [prodlibdb]
GO
EXEC sp_droprole 'ckb_users', 'prodlibuser'
GO
EXEC sp_droprole 'ckb_users'
GO
EXEC sp_addrole 'ckb_users', 'prodlib'
GO
EXEC sp_addrolemember 'ckb_users', 'prodlibuser'
```



J10022

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

### **Steps**

1. Start ArcCatalog.
2. Expand **Database Connections** in the **Catalog Tree** window.
3. Double-click the product library administrator connection geodatabase to connect to it.
4. Select all tables except PCAT\_PERMISSION, right-click, then click **Manage > Privileges**.
5. Type CKB\_USERS into the **User** text box on the **Change Privileges** dialog box.
6. Click the **View (Select)** drop-down arrow and choose **GRANT**.
7. Click the **Edit (Update/Insert/Delete)** drop-down arrow and choose **GRANT**.
8. Click **OK**.
9. Select the **PCAT\_PERMISSION** table, right-click, then click **Privileges**.
10. Type CKB\_USERS into the **User** text box on the **Change Privileges** dialog box.

11. Click the **View** drop-down arrow and choose **GRANT**.

12. 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® software, 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 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.

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 [resources.arcgis.com/en/help/main/10.1/index.html#/What\\_are\\_ArcSDE\\_log\\_file\\_tables/002q00000011000000/](http://resources.arcgis.com/en/help/main/10.1/index.html#/What_are_ArcSDE_log_file_tables/002q00000011000000/).

## Step 13: Create the ArcSDE 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 [prodlibdb]
GO
CREATE USER [prodlibuser] FOR LOGIN [prodlibuser]
GO
```

### Grant privileges:

```
USE [prodlibdb]
GO
EXEC sp_addrolemember N'ckb_users', N'prodlibuser'
GO
```

### Verify user permissions:

```
USE [prodlibdb]
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: Create 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 ArcSDE 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: Assign Product Library Permissions

There are two different levels of interaction with the product library in an ArcSDE 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 [resources.arcgis.com/en/help/main/10.1/index.html#/Product\\_library\\_permissions/010300000043000000/](https://resources.arcgis.com/en/help/main/10.1/index.html#/Product_library_permissions/010300000043000000/).

## Step 16: Add New Users to the Product Library

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, the person must first be added as a product library user, then permissions can be assigned.

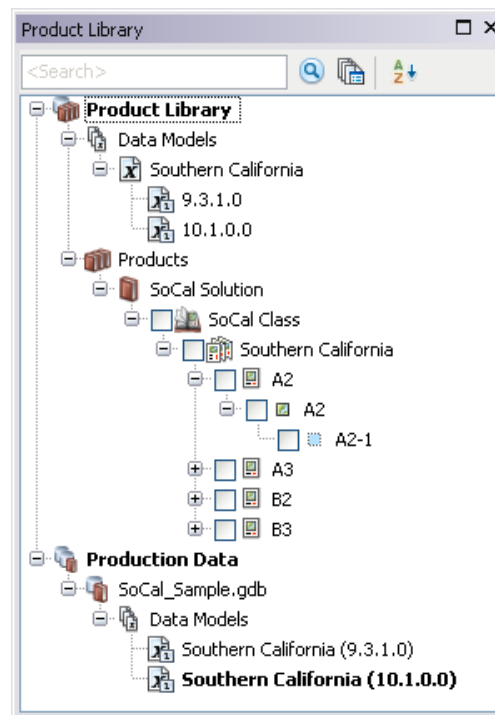
Initially, the user must be added to the geodatabase by the administrator. Each user is added using the first name, last name, and Windows user name.

**Note:** This only applies if you are using an ArcSDE geodatabase as your product library. Personal and file geodatabase permissions are defined by the user's permissions at the operating system level.

### Steps

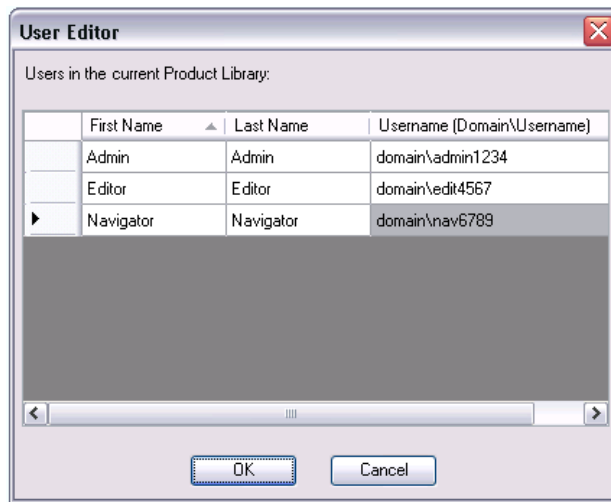
1. Start ArcMap.
2. If necessary, open the **Product Library** window by clicking **Customize > Production > Product Library** on the main menu.

A tree view of the product library appears.



3. If necessary, define the product library workspace.
4. Right-click **Product Library** and click **Configure > 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 to the product library by right-clicking a series and clicking **Permissions**.

5. Right-click anywhere in the **Users in the current Product Library** list and click **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.

6. Type the user's first name in the **First Name** cell.

7. Type the user's last name in the **Last Name** cell.

8. Type the user's Windows login name in the **Username (Domain\Username)** cell.

9. Repeat steps 5 through 8 for each user you need to add to the product library.

10. Click **OK**.

If the user name(s) is valid, the user(s) is added to the product library.

### ***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.

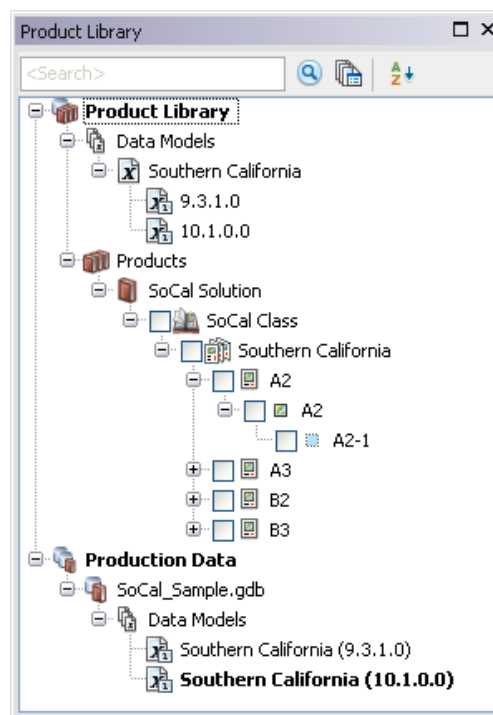
- Check In/Check Out—Files can be checked in and out.
- Edit—Product library levels can be added, modified, and removed, and files can be checked in and out.

**Note:** This only applies if you are using an ArcSDE geodatabase as your product library. Personal and file geodatabase permissions are defined by the user's permissions at the operating system level.

### Steps

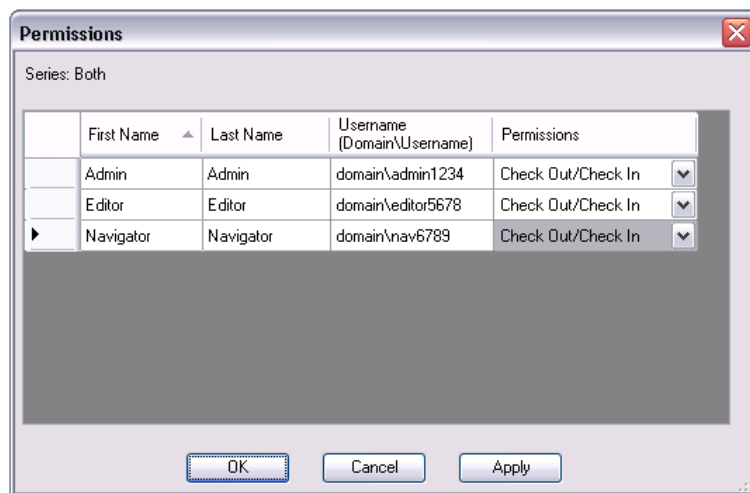
1. Start ArcMap.
2. If necessary, open the **Product Library** window by clicking **Customize > Production > Product Library** on the main menu.

A tree view of the product library appears.



3. If necessary, expand **Product Library** and **Products**.
4. Navigate to the series level of the product class for which you want to assign permissions.
5. Right-click the series name and click **Permissions**.

The **Permissions** dialog box appears.



6. Click the **Permissions** drop-down arrow for the user to whom you want to grant permissions and choose an option.
7. Repeat step 6 for all users to whom you want to give permissions.
8. Click **OK**.

## Replication

You can use geodatabase replication to replicate the Product Library workspace (one-way, two-way replication), but you can create a *new* solution, product class, series, or product only in the parent geodatabase *or* only in the child geodatabase.

## 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:

[resources.arcgis.com/en/help/main/10.1/index.html#/Minimize\\_disk\\_I\\_O\\_contention\\_in\\_SQL\\_Server/002q00000021000000/](https://resources.arcgis.com/en/help/main/10.1/index.html#/Minimize_disk_I_O_contention_in_SQL_Server/002q00000021000000/)

For more information on the product library, visit the Esri Production Mapping page:

[esri.com/software/arcgis/extensions/production-mapping/index.html](https://esri.com/software/arcgis/extensions/production-mapping/index.html)

Learn about setting up the product library in an ArcSDE environment:

[resources.arcgis.com/en/help/main/10.1/index.html#/in\\_an\\_enterprise\\_geodatabase/0103000002sn000000/](https://resources.arcgis.com/en/help/main/10.1/index.html#/in_an_enterprise_geodatabase/0103000002sn000000/)

Access blogs, forums, downloads, and more, via the Esri Production Mapping resource center:

[resources.arcgis.com/en/communities/production-mapping/](https://resources.arcgis.com/en/communities/production-mapping/)

You can access other resources at ArcGIS 10.1 for Desktop Help:

[resources.arcgis.com/en/help/main/10.1/#/Welcome\\_to\\_the\\_ArcGIS\\_Professional\\_Help\\_Library/00qn0000001p000000/](https://resources.arcgis.com/en/help/main/10.1/#/Welcome_to_the_ArcGIS_Professional_Help_Library/00qn0000001p000000/) and Esri Support ([support.esri.com](https://support.esri.com))



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](http://esri.com)

Offices worldwide  
[esri.com/locations](http://esri.com/locations)