

# Best Practices for Storing the ArcGIS® Data Reviewer 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, ArcGIS, ArcSDE, ArcCatalog, ArcMap, [esri.com](http://esri.com), and @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 ArcGIS Data Reviewer Workspace in an Enterprise Geodatabase for SQL Server

## An Esri White Paper

<b>Contents</b>	<b>Page</b>
Introduction.....	1
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 rev User .....	7
Step 3: Modify DBTUNE .....	8
ArcGIS Data Reviewer .....	8
Step 4: Configure SQL Server Parameters .....	8
Step 5: Configure ArcSDE Parameters.....	9
Step 6: Reviewer Workspace .....	9
Step 7: Register the Feature Dataset and Tables as Versioned.....	10
Step 8: Verify Storage.....	10
Step 9: Grant Permissions and Verify Roles.....	11
Tips for Granting Permissions .....	12

<b>Contents</b>	<b>Page</b>
Step 10: Configure Log File Tables .....	12
Step 11: Create ArcSDE Users .....	12
Editor User .....	13
Viewer User .....	13
Replication .....	13
Conclusion .....	13

# Best Practices for Storing the ArcGIS Data Reviewer Workspace in an Enterprise Geodatabase for SQL Server

**Introduction** ArcGIS® Data Reviewer, a data quality control management application, requires you to define a workspace defined when using a review session to store the anomalies found during the data review process. The software creates a new feature dataset and tables in the geodatabase that you have identified as your Reviewer workspace. The feature dataset includes point, line, and polygon feature classes, which contain error geometries for the features that have been written to the Reviewer table.

The Reviewer workspace can be stored in any existing geodatabase, including the production geodatabase, or a separate file or personal geodatabase created specifically for the data review process. When stored in an enterprise geodatabase, the Reviewer workspace supports versioning.

This white paper is intended to help database administrators establish the ArcGIS Data Reviewer 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.

**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 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 Reviewer 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 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 the transaction log files and the data files on separate drives. 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 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

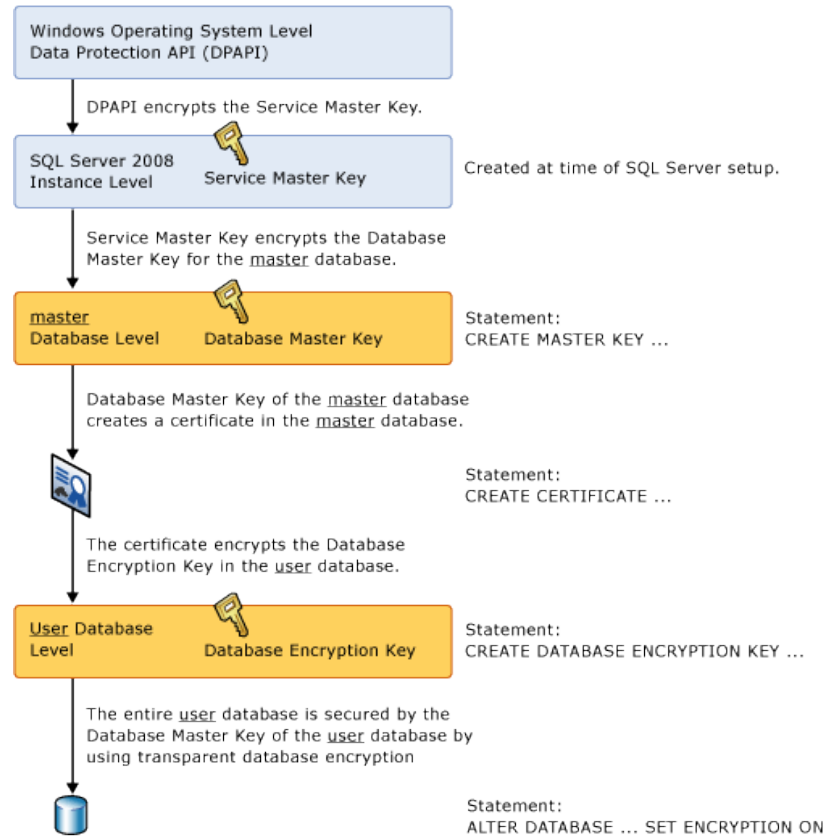
## Transparent Data Encryption

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 the 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.



J9786

```

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

/* Create Certificate */
CREATE CERTIFICATE rev_cert WITH SUBJECT = 'REV Server Certificate';
GO

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

/* Backup certificate */
BACKUP CERTIFICATE rev_cert TO FILE = 'D:\mssqlbackup\master\rev.cert'
WITH PRIVATE KEY (
FILE = 'D:\mssqlbackup\master\rev.pvk',
ENCRYPTION BY PASSWORD = 'rev$$@admin');
GO

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

/* Encrypt database */
ALTER DATABASE revdb 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 = 'revdb'
GO

```

## Step 1: Create Data Files

Create new FileGroups to store the Reviewer feature dataset and tables.

FILEGROUP	ArcSDE_PARAMETER
REV_BDATA	Business table
REV_BINDEX	Business table index
REV_FDATA	Feature table
REV_FINDEX	Feature table index
REV_SDATA	Spatial index table
REV_SINDEX	Spatial index table index
REV_ADATA	Adds table (versioned)
REV_AINDEX	Adds table index
REV_DDATA	Deletes table (versioned)
REV_DINDEX	Deletes table index

```
USE MASTER
GO
ALTER DATABASE [REVDB] ADD FILEGROUP [REV_BDATA]
GO
ALTER DATABASE [REVDB] ADD FILE (NAME = N'rev_Bdata01', FILENAME =
N'C:\mssql\data\revdb\rev_Bdata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[REV_BDATA]
GO
ALTER DATABASE [REVDB] ADD FILEGROUP [REV_BINDE]
GO
ALTER DATABASE [REVDB] ADD FILE (NAME = N'rev_Bindex01', FILENAME =
N'C:\mssql\data\revdb\rev_Bindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[REV_BINDE]
GO
ALTER DATABASE [REVDB] ADD FILEGROUP [REV_FDATA]
GO
ALTER DATABASE [REVDB] ADD FILE (NAME = N'rev_Fdata01', FILENAME =
N'C:\mssql\data\revdb\rev_Fdata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[REV_FDATA]
GO
ALTER DATABASE [REVDB] ADD FILEGROUP [REV_FINDEX]
GO
ALTER DATABASE [REVDB] ADD FILE (NAME = N'rev_Findex01', FILENAME =
N'C:\mssql\data\revdb\rev_Findex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[REV_FINDEX]
GO
ALTER DATABASE [REVDB] ADD FILEGROUP [REV_SDATA]
GO
ALTER DATABASE [REVDB] ADD FILE (NAME = N'rev_Sdata01', FILENAME =
N'C:\mssql\data\revdb\rev_Sdata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[REV_SDATA]
GO
ALTER DATABASE [REVDB] ADD FILEGROUP [REV_SINDEX]
GO
ALTER DATABASE [REVDB] ADD FILE (NAME = N'rev_Sindex01', FILENAME =
N'C:\mssql\data\revdb\rev_Sindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[REV_SINDEX]
GO
ALTER DATABASE [REVDB] ADD FILEGROUP [REV_ADATA]
GO
ALTER DATABASE [REVDB] ADD FILE (NAME = N'rev_Adata01', FILENAME =
N'C:\mssql\data\revdb\rev_Adata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[REV_ADATA]
GO
ALTER DATABASE [REVDB] ADD FILEGROUP [REV_AINDEX]
GO
ALTER DATABASE [REVDB] ADD FILE (NAME = N'rev_Aindex01', FILENAME =
N'C:\mssql\data\revdb\rev_Aindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[REV_AINDEX]
GO
ALTER DATABASE [REVDB] ADD FILEGROUP [REV_DDATA]
GO
ALTER DATABASE [REVDB] ADD FILE (NAME = N'rev_Ddata01', FILENAME =
N'C:\mssql\data\revdb\rev_Ddata01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[REV_DDATA]
GO
ALTER DATABASE [REVDB] ADD FILEGROUP [REV_DINDEX]
GO
ALTER DATABASE [REVDB] ADD FILE (NAME = N'rev_Dindex01', FILENAME =
N'C:\mssql\data\revdb\rev_Dindex01.NDF' , SIZE = 1, MAXSIZE = 800, FILEGROWTH = 1) TO FILEGROUP
[REV_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 [REVDB] MODIFY FILE ( NAME = N'rev_Bdata01', SIZE = 400MB )
ALTER DATABASE [REVDB] MODIFY FILE ( NAME = N'rev_Bindex01', SIZE = 400MB )
ALTER DATABASE [REVDB] MODIFY FILE ( NAME = N'rev_Fdata01', SIZE = 400MB )
ALTER DATABASE [REVDB] MODIFY FILE ( NAME = N'rev_Findex01', SIZE = 400MB )
ALTER DATABASE [REVDB] MODIFY FILE ( NAME = N'rev_Sdata01', SIZE = 400MB )
ALTER DATABASE [REVDB] MODIFY FILE ( NAME = N'rev_Sindex01', SIZE = 400MB )
ALTER DATABASE [REVDB] MODIFY FILE ( NAME = N'rev_Adata01', SIZE = 400MB )
ALTER DATABASE [REVDB] MODIFY FILE ( NAME = N'rev_Aindex01', SIZE = 400MB )
ALTER DATABASE [REVDB] MODIFY FILE ( NAME = N'rev_Ddata01', SIZE = 400MB )
ALTER DATABASE [REVDB] MODIFY FILE ( NAME = N'rev_Dindex01', SIZE = 400MB )
```

### Verify FileGroups and data files:

```
EXEC sp_helpdb revdb
GO
```

J9786

## Step 2: Create rev User

Create a new database user to store the Reviewer feature dataset and tables; grant the appropriate permissions.

### Create user and schema:

```
USE REVDB
GO
CREATE USER [rev] FOR LOGIN [rev]
GO
CREATE SCHEMA [rev] AUTHORIZATION [rev]
GO
ALTER USER [rev] WITH DEFAULT_SCHEMA=[rev]
GO
```

### Grant privileges:

```
USE REVDB
GO
EXEC sp_droprole 'gis_data_creator', 'rev'
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', 'rev'
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) = 'rev'
```

### Associate login rev with user rev:

```
USE REVDB
GO
EXEC sp_change_users_login 'update_one','rev','rev'
GO
EXEC sp_helpuser 'rev'
GO
```

### Step 3: Modify DBTUNE

Export the dbtune file before making any modification.

```
sdeexport -o export -f dbtune_exp.sde -u sde -p sde -i sde:sqlserver:mcsdbsrv -D revdb
```

Copy *dbtune\_exp.sde* to *dbtune\_rev.sde*.

### ArcGIS Data Reviewer

Create a new DBTUNE keyword following the steps below:

- Copy the `##DEFAULTS` keyword and paste it at the end of the *dbtune\_rev.sde* file.
- Rename it `##REVIEWER` and change the FileGroup name for the appropriate parameters.

```
dbtune_rev.sde

##REVIEWER
A_INDEX_RASTER "WITH FILLFACTOR = 75 ON REV_AINDEX"
A_INDEX_ROWID "WITH FILLFACTOR = 75 ON REV_AINDEX"
A_INDEX_SHAPE "WITH FILLFACTOR = 75 ON REV_AINDEX"
A_INDEX_STATEID "WITH FILLFACTOR = 75 ON REV_AINDEX"
A_INDEX_USER "WITH FILLFACTOR = 75 ON REV_AINDEX"
A_INDEX_XML "WITH FILLFACTOR = 75 ON REV_AINDEX"
A_STORAGE "ON REV_ADATA"
B_INDEX_RASTER "WITH FILLFACTOR = 75 ON REV_BINDEX"
B_INDEX_ROWID "WITH FILLFACTOR = 75 ON REV_BINDEX"
B_INDEX_SHAPE "WITH FILLFACTOR = 75 ON REV_BINDEX"
B_INDEX_TO_DATE "WITH FILLFACTOR = 75 ON REV_BINDEX"
B_INDEX_USER "WITH FILLFACTOR = 75 ON REV_BINDEX"
B_INDEX_XML "WITH FILLFACTOR = 75 ON REV_BINDEX"
B_STORAGE "ON REV_BDATA"
D_INDEX_ALL "WITH FILLFACTOR = 75 ON REV_DINDEX"
D_INDEX_DELETED_AT "WITH FILLFACTOR = 75 ON REV_DINDEX"
D_STORAGE "ON REV_DDATA"
F_INDEX_AREA "WITH FILLFACTOR = 75 ON REV_FINDEX"
F_INDEX_FID "WITH FILLFACTOR = 75 ON REV_FINDEX"
F_INDEX_LEN "WITH FILLFACTOR = 75 ON REV_FINDEX"
F_STORAGE "ON REV_FDATA"
GEOMETRY_STORAGE "GEOMETRY"
GEOMTAB_PK "WITH FILLFACTOR = 75 ON REV_FINDEX"
GEOMTAB_STORAGE "ON REV_FDATA"
I_STORAGE "ON REV_FDATA"
S_INDEX_ALL "WITH FILLFACTOR = 75 ON REV_SINDEX"
S_INDEX_SP_FID "WITH FILLFACTOR = 75 ON REV_SINDEX"
S_STORAGE "ON REV_SDATA"
END
```

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

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

```
sdeimport -o import -f dbtune_rev.sde -u sde -p sde -i sde:sqlserver:mcsdbsrv -D revdb
```

### 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 for Reviewer

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

J9786

## Step 5: Configure ArcSDE Parameters

You need to configure the MAXBLOBSIZE and TCPKEEPALIVE parameters for the ArcSDE geodatabase used as the Reviewer workspace. 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.

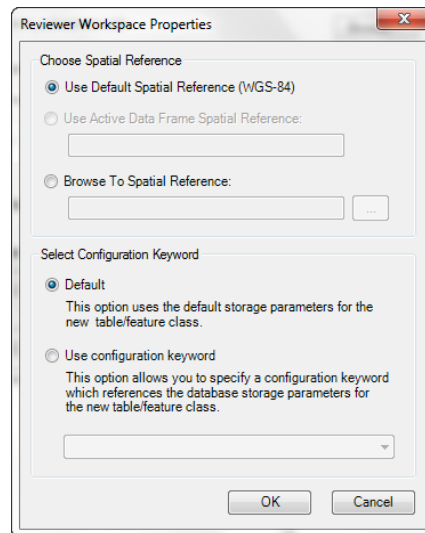
## Step 6: Reviewer Workspace

Create a database connection in ArcCatalog™ with the rev user; this will be the Reviewer workspace location.

In ArcMap™, click the Reviewer Session Manager button on the Data Reviewer toolbar.

- Browse to the Reviewer workspace location.
- Navigate to the database connection created in ArcCatalog.
- Click New to start a new Reviewer session.

The Reviewer Workspace Properties dialog box appears.



*Reviewer Workspace Properties*

- Choose the custom spatial reference.

When you click Browse To Spatial Reference, the New Spatial Reference wizard appears. Choose the spatial reference you want to use with the Reviewer feature dataset.

- Choose the configuration keyword.

You can use the default configuration keyword or select the ##REVIEWER configuration keyword created in step 3.

- Click OK.

This automatically creates the Reviewer feature dataset and tables. The ID and Name fields are automatically populated in the Session area.

- If necessary, type a custom name for the current Reviewer session in the Name text box.

By default, the name matches the ID.

- If you are working with an enterprise database, click the Reviewer Dataset Version drop-down arrow and choose the version to be used.

- Click Start Session.

The button appears to be selected, and its name changes to End Session.

- Click Close.

### Step 7: Register the Feature Dataset and Tables as Versioned

After the new session is created, the rev user must register the following as versioned in ArcCatalog:

- RevDataset (feature dataset)
- RevBatchRunTable table
- RevCheckRunTable table
- RevTableConfig table
- RevTableLocation table
- RevTableMain table

### Step 8: Verify Storage

Run the SQL queries below to verify that the Reviewer dataset was created under the correct FileGroups:

```
USE REVDB
GO
```

#### List FileGroups and data files:

```
EXEC sp_helpdb revdb
GO
```

#### List FileGroup data files:

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

#### List tables by FileGroup:

```
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) = 'rev'
ORDER BY 1,3,2
GO
```

### List indexes by table and FileGroup:

```
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) = 'rev'
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 9: Grant Permissions and Verify Roles

Grant permissions to the rev tables through the schema.

```
USE [revdb]
GO
EXEC sp_droprole 'rev_editor', 'giseditor'
GO
EXEC sp_droprole 'rev_editor'
GO
EXEC sp_addrole 'rev_editor', 'rev'
GO
GRANT DELETE ON SCHEMA::[rev] TO [rev_editor]
GRANT EXECUTE ON SCHEMA::[rev] TO [rev_editor]
GRANT INSERT ON SCHEMA::[rev] TO [rev_editor]
GRANT SELECT ON SCHEMA::[rev] TO [rev_editor]
GRANT UPDATE ON SCHEMA::[rev] TO [rev_editor]
GO
EXEC sp_droprole 'rev_viewer', 'gisviewer'
GO
EXEC sp_droprole 'rev_viewer'
GO
EXEC sp_addrole 'rev_viewer', 'rev'
GO
GRANT SELECT ON SCHEMA::[rev] TO [rev_viewer]
GO
```

### Verify role:

```
EXEC sp_helprole 'rev_editor'
GO
EXEC sp_helprole 'rev_viewer'
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 in ('rev_editor','rev_viewer')  
GO
```

***Tips for Granting  
Permissions***

Grant a user only select, update, insert, and delete privileges to the RevAdminCustomFields and RevAdminDescriptions tables if the user has privileges to modify the database schema and to add customized error descriptions.

**Step 10: 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.

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 11: Create  
ArcSDE Users**

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



J9786

### ***Editor User***

```
USE master
GO
EXEC sp_addlogin N'giseditor', 'gis$editor', @logindb, @loginlang
GO
```

#### **Create user:**

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

#### **Grant privileges:**

```
USE [revdb]
GO
EXEC sp_addrolemember N'rev_editor', N'giseditor'
GO
```

### ***Viewer User***

```
USE master
GO
EXEC sp_addlogin N'gisviewer', 'gis$viewer', @logindb, @loginlang
GO
```

#### **Create user:**

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

#### **Grant privileges:**

```
USE [revdb]
GO
EXEC sp_addrolemember N'rev_viewer', N'gisviewer'
GO
```

### **Replication**

You can use geodatabase replication to replicate the Reviewer workspace (one-way, two-way replication) but you can create *new* Reviewer sessions 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 ArcGIS Data Reviewer workspace 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:

[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 ArcGIS Data Reviewer, visit

[esri.com/software/arcgis/extensions/arcgis-data-reviewer/index.html](https://esri.com/software/arcgis/extensions/arcgis-data-reviewer/index.html) or e-mail [datareviewer@esri.com](mailto:datareviewer@esri.com).

Access blogs, forums, downloads, and more, from the ArcGIS Resource Center:

[resources.arcgis.com/en/communities/data-reviewer/](https://resources.arcgis.com/en/communities/data-reviewer/)

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)