GIS SDE Database Sync



**Table of Contents**

[1 Acronyms 3](#_Toc516483525)

[2 Background 4](#_Toc516483526)

[3 Installation Requirement 4](#_Toc516483527)

[3.1 Privileged Accounts 4](#_Toc516483528)

[3.2 Server with Task Manager Running 4](#_Toc516483529)

[3.3 Enterprise Database 4](#_Toc516483530)

[4 GIS SDE Database Sync 4](#_Toc516483531)

[4.1 Configure the database 4](#_Toc516483532)

[4.1.1 Views 4](#_Toc516483533)

[4.1.2 Tables 4](#_Toc516483534)

[4.2 Configure the Python Script 5](#_Toc516483535)

[4.3 Deploy All Schema SDE Connections 5](#_Toc516483536)

[4.4 Add Database Connections to adminGTS.SDE\_Connections 5](#_Toc516483537)

[4.5 Add Layers To adminGTS.Layer\_Sync\_Control 5](#_Toc516483538)

[5 Troubleshooting 6](#_Toc516483539)

# Acronyms

| **Acronym** | **Description** |
| --- | --- |
| SSMS | SQL Server Management Studio |
| SQL | Structured Query Language |
| T-SQL | Transact-SQL |
| SPROC | Stored Procedure |
| AGOL | ArcGIS Online |
| ESRI | Environmental Systems Research Institute |
| GTS | GIS Technical Services |
| GIS | Geographic Information Systems |
| ITD | Internet Technologies Department |
| SDE | Spatial Data Engine |

# Background

This document covers the installation, configuration and use of the GIS SDE Database Sync python script and associated SQL scripts. The intent of the project was to redevelop the publication scripting tool that had been put into place by a predecessor. There were some areas of interest that merited relooking at how data moves between databases was handled. Some of the more significant changes included the ability to publish only data that had been updated within a certain time window, reducing the need for open accessibility to privileged SDE connections, and the shifting of using the ArcPy libraries for SQL related commands rather than external libraries.

Areas immediately addressed by this script include:

* Publishing on need vs. schedule.
* Reduced configuration needing to be maintained.
* Eliminated 3rd party Python libraries.
* Copious amounts of notes within the Python script.

# Installation Requirement

## Privileged Accounts

You must use a DBO level or higher privileged account to execute portions of this script. The views accessed rely upon areas of the database server that are not accessible under the Public role.

## Server with Task Manager Running

You will need a place to run a scheduled job. Due to that requirement, you’ll also need to install ArcGIS Desktop 10.3.1 or higher. It is also highly recommended you install PyScripter, a freeware application on the server as well to diagnose any problems you may run into.

## Enterprise Database

This script and associated tables and views rely upon a MS SQL enterprise database running ArcSDE. You will need a schema within your source database named adminGTS. This is the schema that is hard coded into the python script as to where the SQL based dependencies exist.

# GIS SDE Database Sync

## Configure the database

### Views

* Open the 3 files located in the root folder DBase Sync/Dependencies/SQL/ named View\_Find\_Layers.sql, View\_Layer\_Table\_History.sql, View\_Layers\_To\_Update.sql with SQL Server Management Studio.
* Execute the three scripts in the following order:
  + View\_Find\_Layers.sql
  + View\_Layer\_Table\_History.sql
  + View\_Layers\_To\_Update.sql

### Tables

* Open the 2 files located located in the root folder DBase Sync/Dependencies/SQL/ named Layer\_Sync\_Control-Table.sql and SDE\_Connections.sql with SQL Server Management Studio.
* Execute the two scripts in any order you prefer.

## Configure the Python Script

* Open the file located in the root folder DBase Sync/ named data\_sync.py with the application named PyScripter.
* Read the top of the file carefully. It contains important information regarding how the script works and what was updated since it was last looked at. It will also talk about how it is to be configured.
* In the section asking for your db\_connection, using ArcCatalog, identify your SDE connection that has enough privileges to read from System Tables and the SDE schema.
  + Update db\_connection to read as db\_connection = r’Database Connections\\YOUR DATABASE CONNEDCTION.sde’
  + Take care to keep the lower case r, the ‘’ as shown and the \\. These are not mistakes and are important to the operation of the script.
* In the section asking for your db\_type, you can put anything you want in there. This really is denoting the environment you are working in. Some call theirs Production, Staging or Test. Other’s can call it something else. Regardless of what you put in here, make sure you include the ‘’ before and after the word. Also, it should not be left blank.

## Deploy All Schema SDE Connections

Make sure you deploy a SDE connection for each Schema you will want to load data against.

## Add Database Connections to adminGTS.SDE\_Connections

Take care to add all your required database connections to adminGTS.SDE\_Connections. Requierd database connections can be defined as any connection necessary to copy data from and/or insert and delete records from.

SourceDB – Name of the database your Connection String references.

SourceDB\_Type – Name of the environment your Connection String references.

Data\_Owner – Schema name of the targeted Connection String.

Conn\_String – Your connection string should include: Database Connections\\ and the name of the SDE connection.

## Add Layers To adminGTS.Layer\_Sync\_Control

Here is a list of fields and what they are used for. Take care to add carefully as the application will do exactly what you ask it to. No more, no less.

ID – This is auto populated.

LayerName – Put the name of your layer in exactly as you want it to look to include Title Casing.

OwnerName – Name of the schema where your layer comes from.

LayerFullName – This field is auto calculated.

FeatureDataSet – Not used at this time.

SourceDB – This is the name of the database you are coming from.

SourceDB\_Type – This is the name of the environment you are taking data from.

ScratchDB – This is the name of the database you will use for re-projecting. If you don’t need to reproject, keep it blank.

ScratchDB\_Type – This is the name of the environment you are using to re-project in. If you don’t need to reproject, keep it blank.

TargetDB – This is the name of the database you are going to.

TargetDB\_Type – This is the name of the environment you are going to.

Process\_Step – Use numbers here. If you are sending a layer to more than one location from your SourceDB, then you need to use process steps. For each process step, you need a number. So you are sending a layer to a database named Utilities and a database named WebGIS you would have a process step 1 and 2. If you get rid of one step, take care to renumber the remaining so you always remain sequential.

Published – a Y or N depending on if you want to publish the layer or not.

Public – a Y or N if the layer is public or not. This does nothing for the script at the moment, but it’s a good idea to know who can access your layers.

Redacted – a Y or N to have the layer redacted. This feature is part of a future enhancement. It does not work presently.

Redacted\_Mask\_Owner\_Name – This feature is part of a future enhancement. It does not work presently.

Redacted\_Mask\_LayerName – This feature is part of a future enhancement. It does not work presently.

# Troubleshooting

Should you find the script is not running as expected, there are but a few areas where things could go wrong. First check the adminGTS.View\_Layers\_To\_Update. This should have only a handful of layers maximum in it every night. If you should find more than that, you know something significant changed in the database. Rebuilding indexes during the day can do that along with normal editing.

The second area to check is to run the Python script on your computer so long as you have all the required connections. If you run into issues, it will error out giving you a general idea as to where the problem is.