













Logging changes to shared databases


 Change logging is only available in SQL Source Control version 3.1 and later.

In the [shared database development model](#), developers work on the same database simultaneously. By default, information about who makes a change to a shared database is read from the default trace and saved in **tempdb**. However, because tempdb resets when the server is restarted, information about who made a change is permanently lost. When this happens, the **Changed by** column lists the object as changed by **Unknown**:

Changed by	Change type	0 of 11 objects with changes to commit
Unknown	 Edit	<input type="checkbox"/> db_ddladmin
Unknown	 Edit	<input type="checkbox"/> noddysview
Unknown	 New	<input type="checkbox"/> newFunc
Unknown	 New	<input type="checkbox"/> NewProcFromSA
Unknown	 New	<input type="checkbox"/> secondUser
Unknown	 New	<input type="checkbox"/> SecondUserView
Unknown	 New	<input type="checkbox"/> Table_1
Unknown	 New	<input type="checkbox"/> Table_3
Unknown	 New	<input type="checkbox"/> Table_4
Unknown	 New	<input type="checkbox"/> Table_5
Unknown	 New	<input type="checkbox"/> Table_6

To prevent this, you can create a new database to permanently log information about changes in. This makes sure information isn't lost, and database administrators can set appropriate security restrictions.

 Setting up a new database to log changes is only necessary when developers share databases. You might want to consider [switching to the dedicated development model](#).

 Information about who made a change can still be lost for other reasons. For more information, see [Object changed by Unknown](#).

Step 1: Creating the change log database

You can use this SQL script template to create a dedicated change log database named ChangeLog. You can modify the script as needed.

SQL script to create ChangeLog database

```
USE master EXECUTE ('CREATE DATABASE ChangeLog')
ALTER DATABASE ChangeLog SET ANSI_NULL_DEFAULT OFF
ALTER DATABASE ChangeLog SET ANSI_NULLS OFF
ALTER DATABASE ChangeLog SET ANSI_PADDING OFF
ALTER DATABASE ChangeLog SET ANSI_WARNINGS OFF
ALTER DATABASE ChangeLog SET ARITHABORT OFF
ALTER DATABASE ChangeLog SET AUTO_CLOSE OFF
ALTER DATABASE ChangeLog SET AUTO_CREATE_STATISTICS ON
ALTER DATABASE ChangeLog SET AUTO_SHRINK OFF
ALTER DATABASE ChangeLog SET AUTO_UPDATE_STATISTICS ON
ALTER DATABASE ChangeLog SET READ_WRITE
ALTER DATABASE ChangeLog SET RECOVERY SIMPLE
ALTER DATABASE ChangeLog SET MULTI_USER
ALTER DATABASE ChangeLog SET PAGE_VERIFY CHECKSUM
ALTER DATABASE ChangeLog SET DB_CHAINING ON
EXECUTE ('USE ChangeLog IF NOT EXISTS (SELECT * FROM sys.sysusers WHERE name='''guest''') EXECUTE
sp_grantdbaccess guest')
```

Step 2: Editing the config file

After the change log database is created, you need to edit a local config file so SQL Source Control can access it.



Every member of your team will have to follow these instructions for SQL Source Control to log their changes.

1. Make sure SQL Server Management Studio is closed.
2. Go to the SQL Source Control config files folder: %localappdata%\Red Gate\SQL Source Control 3
Open *RedGate_SQLSourceControl_Engine_EngineOptions.xml* in a text editor.
3. Below the `EngineOptions version` line, add:

```
<TraceCacheDatabase>ChangeLog</TraceCacheDatabase>
```



The file is case sensitive. Don't change the capitalization of the text.

Ignoring any comments (indicated with `<!-->`), the final file should look like this::

Example

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
  <EngineOptions version="3" type="EngineOptions">
    <TraceCacheDatabase>ChangeLog</TraceCacheDatabase>
  </EngineOptions>
```



The example above doesn't include any extra lines you may have included for other options.

4. Save and close the file.

SQL Source Control will now use the change log database to log changes made to all linked databases.



- Each developer must have `dbo_owner` permissions for the change log database.
- You can delete the change log database, but history about changes will be permanently deleted.
- SQL Source Control will only use the change log database to save information about changes to linked databases. It won't be used for any other purpose.



The change log database will contain details of changes made to all databases linked to SQL Source Control. Users will see the names of modified objects, but not the data itself. If the object names in your database contain sensitive information, consider restricting access instead of using the guest role.

How to check if changes are being logged

To check if change logging is set up correctly, interact with a linked database.

Afterwards, in the ChangeLog database, the table `RG_AllObjects` should appear. You can inspect the table to see changes appearing in it as you make them.

Object types not supported by change logging

Changes made to the following object types can't be logged:

application roles	extended properties	search property lists
asymmetric keys	full text catalogs	symmetric keys
certificates	full text stoplists	table keys
constraints	partition functions	user-defined data types
DDL triggers	partition schemes	user-defined table types
DML triggers	roles	user-defined types
event notifications	schemas	users



Changes made to [data](#) can't be logged.