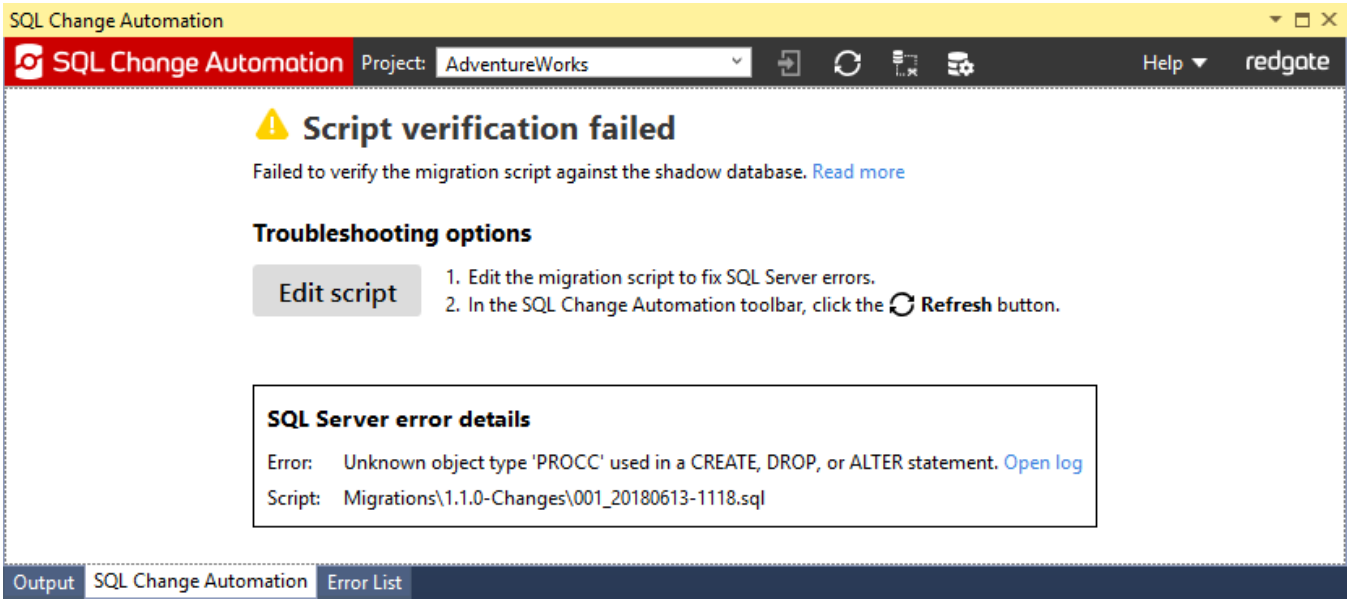
Script verification

Every time you use the SQL Change Automation window to import a change, SQL Change Automation generates a migration script.

When you click **Refresh (Verify Script)**, SQL Change Automation validates the script by running it against the shadow database. This identifies problems before the script is deployed.

If a script fails verification, this view is shown:



To view the full details of the error, under **SQL Server error details**, click the **Open log** hyperlink.

The **Undo import** option isn't available if you've created the script manually, or it was generated on someone else's SQL Server instance. This is because there's no earlier version to revert to.

Fixing script verification failures

The verification error dialog explains how to alter scripts in order to fix issues with them. The *Script verification failed* view shows one SQL Server error at a time. If the script contains multiple errors, it'll keep failing verification until every error is fixed. To resolve the error by hand, click *Edit script* to open the script in the document window and make any necessary adjustments.

If you continue to receive errors, in spite of having fixed the problems in the affected scripts, try forcing a rebuild of the shadow database. This can be done by executing the *Build... Clean Solution* command in Visual Studio (which will drop the shadow db from your development instance) and then clicking **Refre sh** within the SQL Change Automation tool-window.

The verification process can catch common issues before the script is deployed. For instance:

- Missing dependencies
    - A user that does not have a related login
    - A stored procedure that depends on a full-text index being present
    - A view that is missing a dependent view
    - Unsupported objects due to dependency chains
- Invalid code in legacy objects
    - Stored procedures that still use old-style outer joins
- Syntax errors (only possible if the script was manually edited)

Excluding objects from script verification

In some situations, it may be desirable to exclude certain objects or T-SQL batches from the verification process. In particular, if your project scripts make changes to server-level objects, such as linked-servers or SQL agent jobs, having them execute during the shadow database deployment may result in those objects being overwritten. It may also be desirable to exclude certain cross-database operations from verification, such as an UPDATE statement that selects data from another database.

This can be achieved by gating your batches with the *$(IsShadowDeployment)* system variable:

```
IF '$(IsShadowDeployment)' = 0
        BEGIN
                PRINT N'Creating linked-server object [LinkedServerAlias] because IsShadowDeployment=0...';
                EXEC sp_addlinkedserver @server=N'LinkedServerAlias'
                , @srvproduct=N'sql_server'
                , @provider=N'SQLOLEDB'
                , @datasrc=N'ActualServerName'
        END
ELSE
        BEGIN
                PRINT N'Skipping linked-server object [LinkedServerAlias] creation because
IsShadowDeployment=1';
        END
GO
```

Although it is technically possible to use the above method to exclude DDL statements from the verification process, we would not recommend excluding database-level objects (such as CREATE/ALTER TABLE statements) within the shadow database deployment. Doing so may impact the integrity of the script generation process, as it may lead SQL Change Automation's comparison engine to make incorrect assumptions about the current state of your schema model.