Working with existing databases

Importing existing database objects

To import objects from an existing database:

1. Create a new ReadyRoll Database project/solution. Once the project is created, the ReadyRoll window should appear.

   If the ReadyRoll window does not automatically appear, from the View menu in Visual Studio, select ReadyRoll.

2. Step through the wizard to connect to the database you would like to import.

   As ReadyRoll creates objects within your database during import (a table called [dbo].[__MigrationLog] plus a view called [dbo].[__MigrationLogCurrent]), we recommend against connecting your project directly to a Production instance of SQL Server to avoid any potential impact to your business systems.

   Instead, we recommend taking a backup of your Production database, removing any sensitive data, and restoring it onto a Test or Development SQL Server that you can import from.

3. Click Import to generate a baseline script for the existing objects within the database.

   The wizard will import the entire schema within the database, generating scripts for all of the contained objects. If this is what you want, then you can skip straight to step 6 to verify the imported migration.

   On the other hand, if you only want to import a sub-set of objects, go ahead and delete the migration that was added to the project by this step (within the Migrations sub-folder) and continue to the next step to de-select the objects that you do not want to import.

4. Click Refresh from the toolbar of the ReadyRoll window. This scans the connected database for objects that are pending import:

5. Deselect objects that you don't want to import.

6. To create a new migration containing the existing objects, click Import (Generate Script). The ReadyRoll window will update, showing a summary of the objects imported:
6. To test the deployment of the newly-generated script against the Shadow database, click Refresh. For more information about the shadow database, see Target and shadow databases.

This set of steps will allow you to reproduce the "shell" of your database from scratch; you will be able to deploy your tables, views, procedures etc to a new database, however it will not include any data.

For information on how to populate your database with real-world data (including reference/lookup/static data), see this article on data population techniques.

Preparing the database project for your first deployment

Before you can start deploying your database project to a pre-existing database, you need to set a baseline within the database. A baseline is a "line in the sand" between the current state of your existing Production database and any new changes to be deployed. If your Production database does not exist yet, then you do not need to set a baseline.

There are a number of different ways that you can set up the database project for your first ReadyRoll deployment. However what actually happens when you use these different methods is basically the same:

1. ReadyRoll creates two objects in your database:
   - **(Table) [dbo].[__MigrationLog]** keeps track of the migrations and programmable object/additional scripts that have been executed against your database (additional executions of programmable objects/additional scripts will result in new rows being inserted)
   - **(View) [dbo].[__MigrationLogCurrent]** lists the latest version of each migration/programmable object/additional script to have been executed against the database

2. ReadyRoll inserts row(s) into the [dbo].[__MigrationLog] table:
   - If the server method is used to set the baseline, a row will be inserted for each file within the baseline folder (i.e. Migrations\1.0\0-Baseline\*.sql). Additionally, if the programmable objects option is checked, a row will be inserted for each file within your programmable objects folder (i.e. Programable Objects\**\*.sql)
   - If the Visual Studio method is used, a single row will be inserted for the migration open in the document window (e.g. Migrations\0001_Imported-Objects.sql)
   - If the command line method is used, one or more rows will be inserted (i.e. depending on what you supply to the "baseline up to migration" argument)
   - If you connected your ReadyRoll database project directly to your Production environment and clicked Import, then a row will be inserted for each migration and programmable object that was generated by that import.

   **What happens when you import directly from Production**

   If you followed the last method (connected your project directly to Production and imported migrations) then your database is already set up for the first deployment: no further action is needed, as your Production environment was already populated with the required migration log records when you clicked Import. Once you've authored some new migrations, you'll be ready to perform your first deployment.

Ensuring that new db changes are included in deployments

When deciding which migrations to include in the baseline, firstly ensure that the T-SQL logic to reproduce your Production database is
SemVer method (recommended)

If you have many pre-existing databases in your environments, to save time you may like to consider using the semver method to set the baseline. This method actually stores the baseline within the database project itself, effectively making the act of setting the baseline a one-time operation.

For detailed information on how this method works, see *setting the baseline with semantic versioning*.

To use the semver method to configure the baseline for your project: select the **Set the baseline for existing databases using the first semver folder** checkbox within the project designer (Project Settings tab). During the first deployment to a pre-existing database, this will cause ReadyRoll to mark-as-deployed any scripts have been put into the *Migrations\1.0.0-Baseline* folder, thus preventing the scripts from being executed. Any scripts placed within subsequent semver folders (e.g. *1.1.0-NewChanges*) will then subsequently be executed. This folder will be executed.

Pre-deployment baseline check
If you attempt to deploy your database project (or a package created from your database project) to an existing database prior to configuring a baseline, ReadyRoll will raise an error and abort the deployment before any scripts are executed. This is to ensure that you do not accidentally deploy over the top an existing database. To override this error and force the deployment to continue, specify the $ForceDeployWithoutBaseline = True variable in PowerShell (or add ForceDeployWithoutBaseline=True as a variable within your Octopus Deploy project).

If you have enabled the **programmable objects feature** in your project, it is possible to also exclude those from being executed during the first deployment to a pre-existing database. To mark-as-deployed any scripts within the Programmable Objects sub-folder, select the **Mark all programmable objects/additional scripts as deployed when setting the baseline** checkbox (this can be found under the Additional Scripts section of the Project Settings tab).
Visual Studio method

To set a baseline within Visual Studio, connect the project to the existing database and open the imported object script for editing.

Click **Mark as Deployed** to insert the migration log record. Repeat this process for any other scripts that were imported from your Production database.

When you deploy the database project, the scripts that you marked as deployed will not be executed.
Command line method

To set a baseline at the command line, use the MSBuild tool to deploy the migration log objects and insert the baseline row(s). Pass in a value for the /p:BaselineUptoMigrationId= argument containing the name of the first migration within your project (e.g. 0001_Imported-Objects.sql):

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe
AdventureWorks.sqlproj
/p:DBDeployOnBuild=True
/p:TargetServer=MYDBSERVER
/p:BaselineUptoMigrationId=0001_Imported-Objects.sql
```

Example

Including multiple scripts in the baseline

If you created multiple scripts when you imported your database, specify the last script that was imported/added to the project for the BaselineUptoMigrationId property value. For example, if you specify /p:BaselineUptoMigrationId=003_Imported-Data.sql, scripts 001, 002 and 003 will be included in the baseline.

Authentication mode

By default, ReadyRoll will use Windows Authentication to connect to your database. If you would like to use SQL Server authentication instead, specify the /p:TargetUserName=... and /p:TargetPassword=... command line arguments.