

ECO quick start 08 - Database operations

Table of Contents

Overview	1
Prerequisites and goals	2
Evolving the database	3
Reverse engineering an existing database	5
Mapping a model to an existing database	9
Index	a

1 Overview

2 Prerequisites and goals

Prerequisites

To successfully follow this document the user should have read the preceding articles in this series and have a saved project.

Goals

By the end of this document you be able to

- Evolve a database to accommodate changes to your business model.
- Reverse engineer an existing database.
- Map a model to an existing database.

3 Evolving the database

Once your application is finished there will undoubtedly come a time when modifications are required. ECO has the ability to take a database created with an older version of your software and to upgrade or "evolve" it to suit the new business needs of a later release. Modifications are made to the database structure whilst preserving data, supported data migrations are:

- Renaming classes.
- Renaming class attributes.
- Renaming associations to other classes.
- Renaming link tables used in many-many class associations.
- Inserting a new superclass.
- Removing a superclass.
- Moving attributes and associations up or down the class hierarchy.

Note: To rename a member simply change its name and then record its old name in its "FormerNames" property.

Examples of changes to the model that are not supported are:

- Changing the type of an attribute.
- Adding and removing NOT NULL constraints.
- Changing whether or not an attribute has special behavior such as triggers for auto-increment support.
- Changing the multiplicity of an association from single to multiple, or multiple to single.
- Changing which table in a 1..1 association holds the ID to the other table.

Although it is possible to invoke the auto-evolve feature of ECO at runtime on a live system it is recommended that you instead use a SQL script to evolve the live database, it is also good practise to backup your live database before performing any structural changes whether using ECO or not. ECO can be used to generate the SQL script required to evolve your database to support the latest version of your business model, you may then review the script and make any adjustments you feel are appropriate before executing it on the live database.

1. Ensure the ECOWinFormDemo project is open.
2. Compile the project to ensure the binaries are up to date with the latest model information.
3. On the bottom of the EcoSpace select the "Evolve Schema" button.



4. A form will now appear explaining the schema evolve steps to be taken.

```

Detecting changes
Initializing Script
Detecting type clashes...
Analyzing new tables...
Analyzing new columns...
Analyzing new instances...
Analyzing data movements...
Analyzing old instances...
Analyzing old columns...

```

```
Analyzing old tables...
Optimizing script...
Add table Category
Add column Article.ReviewState
Add column Article.MainState
Add column Article.ReasonRejected
Add column Author.AutoPublishArticles
Add column Article.Category
```

5. The [SQL script] tab contains the SQL script required to upgrade the database.
6. The [Mapping info] tab contains the object to table mapping information.
7. Click the "Execute" button and all necessary changes will be made to the development database.

4 Reverse engineering an existing database

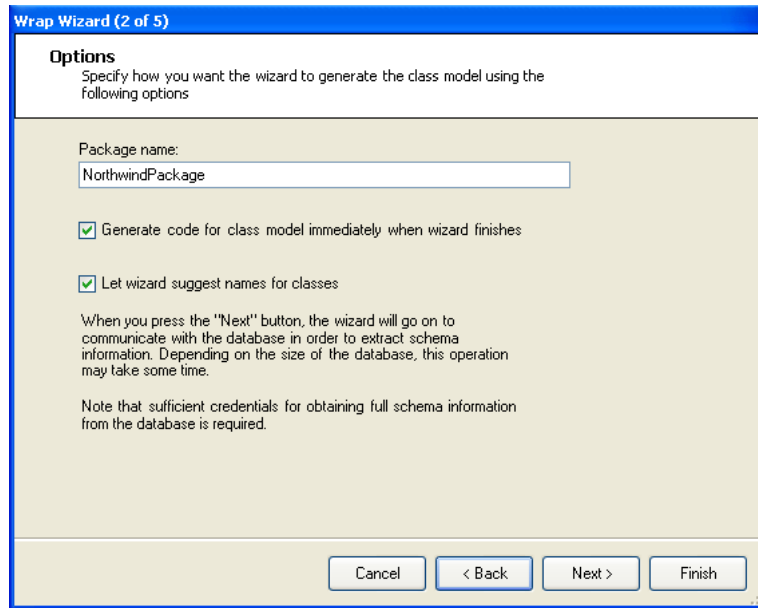
1. Create a new "ECO Package in a Package" (Delphi for .NET) or "ECO Package in a DLL" (C#).
2. Remove the package source code + the EcoPkg files from the project by right-clicking them in the project manager window.
3. In the [Model View] tab remove the package node and package source code by right-clicking each in turn and selecting "Delete".
4. The project is now a package-less project but with all of the .NET DLL references it requires to compile when a new package is added automatically by ECO at the end of the reverse engineering process.

Next a connection to the database is required in order for the ECO wizard to convert it into a model. To do this an EcoSpace is required, once the wizard is complete it is okay to delete this EcoSpace.

5. Select the File->New->Other menu.
6. From the relevant treeview node on the left (C# or Delphi for .NET) select "New ECO files".
7. From the icons on the right select "EcoSpace".
8. Click the "OK" button.
9. Add a SqlConnection component to the EcoSpace and configure its connection string so that it points to the Northwind database installed with SQL Server by default.
10. Add a PersistenceMapperSqlServer to the EcoSpace and set its "Connection" property.
11. At the bottom of the Object Inspector click the option "SQL Server Setup" to configure the persistence mapper component.
12. Select the EcoSpace and ensure its "PersistenceMapper" property is set to the PersistenceMapperSqlServer.
13. Compile the application, this creates the binaries required by ECO.
14. At the bottom of the EcoSpace click the icon "Wrap existing database with ECO".

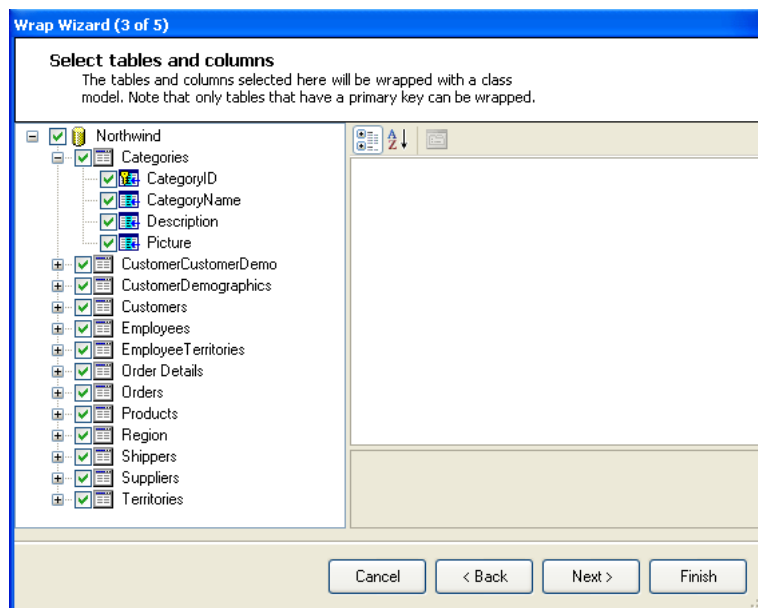


15. In the first step of the wizard click "Next".
16. In the second step the wizard enter "NorthwindPackage" as the package name.
17. Leave the default options checked in the checkboxes below the package name text box.



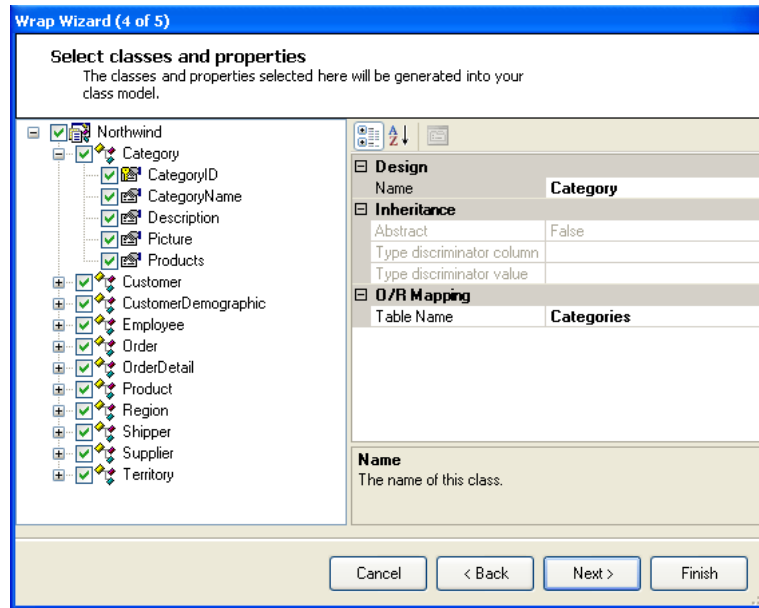
18. Click "Next".

19. The next page in the wizard will show a list of tables in the Northwind database, leave them all selected. Here it is possible to expand each table and select or deselect individual columns.



20. Click "Next".

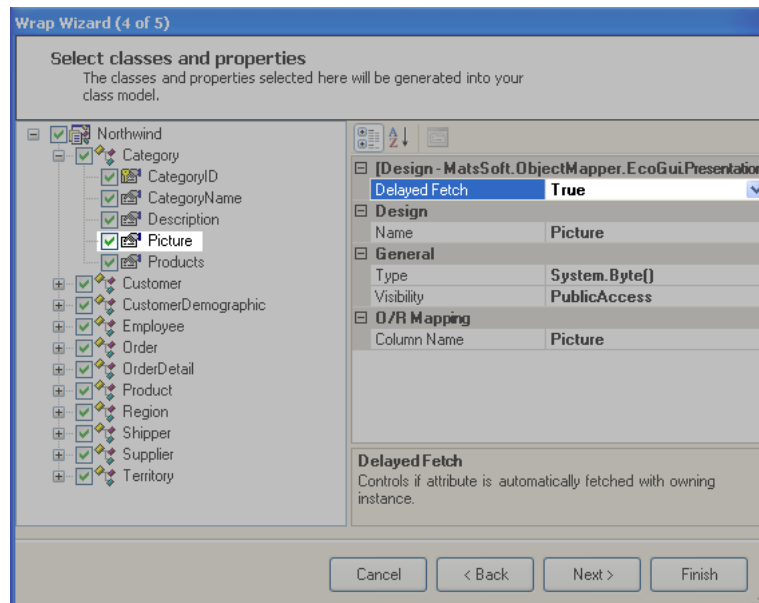
21. The next step in the wizard will show a list of classes to be generated from the tables. Note how the plural table names such as "Categories" are converted to singular class names such as "Category".



22. Expand the Category class in the treeview.

23. Select the "Picture" node.

24. Set its "DelayedFetch" property to "True", this will ensure that the Category's "Picture" will only be fetched from the DB when an attempt is made to read Category.Picture.



25. In the final step of the wizard click "Finish".

26. You are now presented with an XML file in the editor, this contains the class to RDBMS mapping information in XML format. Click "Save all" in the IDE.

27. //TODO: Close the project and reopen it (or) Click "Refresh model view" in the [Model View] tab.

28. Double-click the "NorthwindPackage" node in the [Model View] tab, this will open up a class diagram.

29. To tidy up the diagram layout right-click an empty area of the diagram and select "Layout" and then "Do full layout" from the context menu.

30. Compile the project and save all changes.

The XML generated by the wizard contains all of the information required to map the modeled business classes to the existing database. The section "Mapping a model to an existing database ([see page 9](#))" explains how to use this XML file.

5 Mapping a model to an existing database

In addition to being able to generate a database structure to persist modeled business classes, ECO is also capable of persisting to either an existing database or to a user defined database structure.

1. Create a new ECO WinForms application.
2. Add the generated DLL file from "Reverse engineering an existing database (see page 5)" to the project's "References" list.
3. Compile the application to update the project binary files.
4. Double-click the EcoSpace file in the project manager, and select the [Design] tab in the editor.
5. Click the "Select packages" icon on the toolbar at the bottom of the EcoSpace.



6. Remove the default package from the "Selected" list on the right.
7. Add the NorthwindPackage from the "Available" list on the left.
8. Click "OK"
9. Add a SqlConnection component to the EcoSpace and set its ConnectionString to point to the Northwind database.
10. Add a PersistenceMapperSqlServer to the EcoSpace.
11. In the Object Inspector (or right-click context menu) for the component select "SQL server setup".
12. Set its "Connection" property to "SqlConnection1".
13. Set the "PersistenceMapper" property of the EcoSpace to "PersistenceMapperSqlServer1".

Next some custom mapping information will be connected to the persistence mapper.

14. Add a FileMappingProvider to the EcoSpace.
 15. Set its "FileName" property to point to the XML file generated by "Reverse engineering an existing database (see page 5)".
 16. On the PersistenceMapperSqlServer1 component set the "RuntimeMappingProvider" to point to the FileMappingProvider.
- Finally a very simple test application will be developed to demonstrate data being fetched from the database.

17. Open the WinForm in the designer by double-clicking it in the project manager.
18. Select the rhRoot component.
19. In the Context Agent expand the "Category" node and then drag the "Category.allInstances + DataGrid" node and drop it onto the form.
20. Run the application and you will see a list of Category objects retrieved from the database.

Index

E

Evolving the database 3

M

Mapping a model to an existing database 9

O

Overview 1

P

Prerequisites and goals 2

R

Reverse engineering an existing database 5